# TABLE OF CONTENTS


| Chapter | Title |
|---|---|
| Chapter 1 | Introduction: Three Industries, One Mathematics |
| Chapter 2 | Market Analysis: OTT, Banking & FinTech Landscape |
| Chapter 3 | Unit 1 — Sets, Relations & Functions |
| Chapter 4 | Unit 2 — Counting Techniques & Combinatorics |
| Chapter 5 | Unit 3 — Propositional Logic & Proof Techniques |
| Chapter 6 | Unit 4 — Algebraic Structures & Morphism |
| Chapter 7 | Unit 5 — Graphs and Trees |
| Chapter 8 | Detailed Case Studies with data |
| Chapter 9 | Visual Summary Concept-Industries Mapping |
| Chapter 10 | Step-By-Step Worked Examples |
| Chapter 11 | Conclusion, Future Scope & Summary Tables |
| References | Research Papers, Books, Reports |

# CHAPTER 1: INTRODUCTION -THREE INDUSTRIES, ONE MATHEMATICS

## 1.1 The Digital Economy and Mathematical Foundations

We live in an era where three industries — media streaming, banking, and financial technology — collectively touch the lives of over four billion people every single day. A teenager in Mumbai opens Netflix and is immediately shown a perfectly curated list of shows. A merchant in Jaipur receives a payment in under two seconds through UPI. A bank's fraud system silently evaluates 3,000 data points about a credit card transaction in 50 milliseconds and decides whether to allow it. These are not miracles of pure engineering — they are applied mathematics.

Discrete Mathematics forms the theoretical bedrock of computer science. It provides the vocabulary (sets, functions, relations), the logic (propositional and predicate calculus), the counting tools (permutations, combinations, induction), the structural frameworks (groups, rings, boolean algebras), and the network models (graphs and trees) that every software system ultimately depends on. The goal of this assignment is to make those connections explicit, concrete, and illustrated with real data.

## 1.2 Three Industries at a Glance

### 1.2.1 OTT Platforms — The Recommendation Economy

Over-the-Top (OTT) platforms deliver audio-visual content via the internet, bypassing traditional broadcast and cable television. Netflix, Amazon Prime Video, Disney+ Hotstar, YouTube Premium, Apple TV+, JioCinema, SonyLIV, Zee5 — these are the defining media companies of the 21st century. The industry's defining challenge is discovery: with thousands of titles and billions of users, how does each person find something to watch in under 90 seconds?

The answer is algorithmic recommendation — a system built on functions, graphs, matrix algebra, probabilistic inference, and combinatorial search. Netflix has publicly stated that its recommendation engine saves over USD 1 billion per year by reducing subscriber churn. Approximately 75–80% of all viewing hours on Netflix come from algorithmic recommendations rather than user searches.

**Key OTT Facts (2024–2025)**

Global OTT video revenue: USD 316 billion in 2024 → USD 595 billion by 2030 (CAGR 16.7%)
Global OTT user base: 3.71 billion in 2024, expected 5.1 billion by 2030
Netflix subscribers: 301.6 million (Q3 2025); Q1 2025 revenue: USD 10.54 billion
US market: 99% of households subscribe to at least one OTT service (avg 2.9 subscriptions)
Average American streams 17–21 hours of OTT content per week
Disney+ Hotstar: 59 million peak concurrent viewers during 2023 Cricket World Cup Final
Recommendation algorithms contribute 75–80% of viewing hours on Netflix
Netflix A/B testing: ~1,000 experiments/year; thumbnail tests alone raised CTR by 20–30%

### 1.2.2 Banking Systems — Trust at Scale

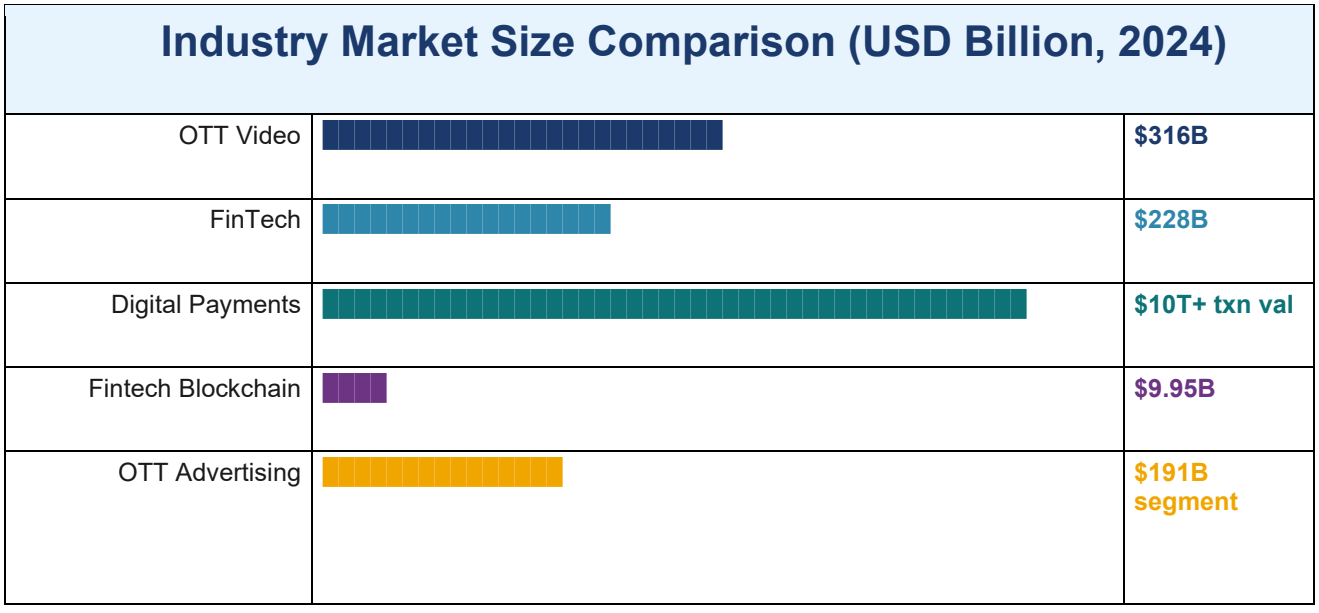
The modern bank is fundamentally a software company that holds deposits. The State Bank of India alone has over 500 million account holders — more people than the entire population of the United States of America. Every ATM withdrawal, every NEFT transfer, every credit card swipe, every UPI payment triggers a cascade of software operations: authentication, authorization, account lookup, fraud scoring, transaction recording, reconciliation, and regulatory reporting.

Machine learning is now embedded at every layer of banking. In fraud detection alone, graph neural networks (GNNs) operating on transaction graphs have been shown to outperform traditional rule-based systems by 5–6 times. A 2025 study published in Cogent Business & Management trained models on 565,000 real banking transactions and achieved 95.79% accuracy for fraud detection using Random Forest — a tree-based ensemble method rooted in discrete mathematics.

### 1.2.3 FinTech — Disrupting Finance with Mathematics

FinTech companies use technology to deliver financial services cheaper, faster, and more inclusively than traditional banks. India's Unified Payments Interface (UPI) — built by NPCI — is the world's most advanced real-time payment infrastructure. In 2025, UPI processed 228.3 billion transactions worth ₹299.7 lakh crore (~USD 3.6 trillion). In December 2025, UPI set a record of 21.63 billion transactions in a single month.

Globally, FinTech companies raised USD 43.5 billion in investment in 2024, and as of January 2025, there were 414 FinTech unicorn companies worldwide. The underlying mathematics is profound: cryptographic protocols protect billions of transactions daily using group theory and number theory; Merkle tree structures power blockchain ledgers; routing algorithms based on shortest-path graph theory direct funds across payment networks in milliseconds.



# CHAPTER 2: MARKET ANALYSIS — OTT, BANKING & FINTECH

## 2.1 Global OTT Platform Market Deep Dive

The global OTT market has become one of the fastest-growing sectors in the technology landscape. According to Grand View Research, the market was valued at USD 298.84 billion in 2024 and is projected to reach USD 1,425.63 billion by 2033, growing at a CAGR of 18.6%. Precedence Research places the 2034 projection even higher — at USD 2,816.85 billion — with a CAGR of 24.42%. The variance between forecasts reflects differences in scope (devices vs. services only) but all agree: this is one of the fastest-growing markets in history.

| Metric | 2024 Value | 2030 Projection | 2033-34 Projection | CAGR |
|---|---|---|---|---|
| Global OTT Revenue | $316B (Statista) | $595B | $1,425–2,817B | 16.7–24.4% |
| OTT Users Worldwide | 3.71 Billion | 5.1 Billion | Not specified | ~6% |
| ARPU (OTT Video) | $85.04 | ~$95 | ~$110 | ~2.5% |
| OTT Advertising Segment | $191.4B | $207.5B | $400B+ | 10.3% |
| Video Streaming (SVOD) | $108.5B | $119.1B (2025) | $300B+ | 13.2% |
| North America Share | 40% (largest) | 38% | 35% (declining) | — |
| Asia-Pacific OTT | Fastest growing | 2nd largest | 1st by 2030 | 22%+ |

### 1.1 OTT Platform User Behaviour Statistics

Understanding how users interact with OTT platforms drives every engineering and product decision, including those based on discrete mathematics. The following statistics, sourced from Statista, Uscreen, and industry surveys, define the user behaviour landscape that recommendation algorithms must serve:

- 70%+ of OTT users binge-watch content (consume multiple episodes in one session)
- 34.97% of users stream OTT content multiple times per day
- 74.89% of users access OTT platforms at least three times per week
- 65% of streaming is done via mobile and TV apps rather than browsers
- Average American spends 70 minutes per day on OTT platforms
- 99% of US households have at least one OTT subscription (avg 2.9 per household)
- Netflix: 75–80% of all viewing hours come from algorithmic recommendations
- Netflix saves USD 1+ billion/year through personalization (reduced churn)

## 2.1.2 Platform-Specific Data

| Platform | Subscribers/Users | 2025 Revenue | Key AI Feature | Discrete Math Used |
|---|---|---|---|---|
| Netflix | 301.6M (Q3 2025) | $10.54B (Q1 2025) | Personalized homepage, artwork selection | Graph theory, sets, matrix algebra |
| Amazon Prime Video | 200M+ (est.) | Part of $600B+ AWS revenue | X-Ray metadata, watchlists | Set operations, graph databases |
| Disney+ (incl. Hotstar) | 153.8M subscribers | $24B Disney+ revenue | Sports streaming, live events | Counting, combinatorics, probability |
| Apple TV+ | 45M+ (2025) | ~$5B est. | Siri recommendations | NLP, set theory |
| YouTube (Premium) | 2B+ users total | $8.9B (Q4 2024) | Auto-play, comment ranking | PageRank graph algorithm |
| JioCinema/Hotstar India | 500M+ MAU | Merged Reliance-Disney ($8.5B deal) | IPL live streaming records | Connectivity algorithms |

## 2.2 Global Banking Sector Analysis

The banking sector is the world's most data-intensive industry. A single large bank may process 30–50 million transactions per day. Globally, over 1 billion banking transactions are processed daily. The consequences of errors are measured in billions of dollars and in systemic financial risk. This creates enormous demand for mathematically guaranteed algorithms — systems whose correctness can be formally proved, not just empirically tested.

## 2.3 Global FinTech Market Analysis

FinTech occupies the intersection of finance and technology — and increasingly, of finance and mathematics. The global FinTech market reached USD 228.02 billion in 2024 (Zion Market Research) and is projected to grow to USD 727.17 billion by 2034 at a CAGR of 15.60%. Fortune Business Insights places the 2025 valuation at USD 394.88 billion, projecting USD 1,760.18 billion by 2034 at a CAGR of 18.20% — one of the highest CAGRs of any major industry sector.

| FinTech Sub-Sector Market Sizes (2024–2025) | | |
|---|---|---|
| Overall FinTech | | **$228–395B** |
| Digital Payments | | **$10T+ (txn val)** |
| Blockchain in FinTech | | **$9.95B** |
| FinTech Blockchain (adj) | | **$0.48T** |
| Digital Lending | | **$567B by 2026** |
| WealthTech / Robo-Advisory | | **$100B+** |

| FinTech Segment | 2024 Value | Growth Driver | Discrete Math Application |
|---|---|---|---|
| UPI (India) | 228.3B txns (2025) | Smartphone + internet penetration | Graph routing, combinatorics, modular arithmetic |
| Blockchain | $9.95B market | DeFi, CBDCs, cross-border | Group theory, Merkle trees, cryptographic hashing |
| AI in FinTech | Largest CAGR sub-segment | Fraud, credit scoring, chatbots | Decision trees, Boolean algebra, graph neural networks |
| Digital Wallets | $20.37T txn value (2025) | Mobile-first economies | Set theory, equivalence classes, tokenization bijections |
| Open Banking/APIs | $43.15B by 2026 | Regulatory mandates (PSD2, RBI) | Homomorphisms, schema graphs, formal verification |
| BNPL | $700B+ by 2028 | Gen Z preference | Combinatorics, credit scoring trees, risk probability |

CHAPTER 3: UNIT 1 — SETS, RELATIONS & FUNCTIONS

## 3.1 Introduction to Unit 1 Concepts

Sets, relations, and functions form the language of mathematics — and therefore the language of software. A database table is a set of tuples. A REST API is a function. A user's subscription history is a relation between users and content. The elegance of set theory is that it provides a single unified framework for all of these, and the tools built on top of it (cardinality, power sets, bijections, partial orders) translate directly into engineering design patterns.

**Unit 1 Syllabus Topics Covered**

1. Operations and Laws of Sets
2. Cartesian Products
3. Binary Relation
4. Partial Ordering Relations
5. Equivalence Relation
6. Image of a Set
7. Bijective Functions
8. Inverse and Composite Functions
9. Size of a Set
10. Finite and Infinite Sets
11. Countable and Uncountable Sets
12. Cantor's Diagonal Argument
13. The Power Set Theorem
14. Schroeder–Bernstein Theorem

## 3.2 Sets and Set Operations in OTT Platforms

### 3.2.1 User Segmentation — Industrial Application of Set Theory

Netflix and every major OTT platform begin recommendation by partitioning users into meaningful groups. This is set theory made operational. Define the universal set U as all active subscribers. Then create subsets based on viewing behaviour, demographics, and preferences. The diagram below shows how Netflix segments users using set operations:

| **Netflix User Segmentation Flow (Set Theory)** |
|---|
| **STEP 1:** Define U = all 301.6 million Netflix subscribers |
| ▼ |
| **STEP 2:** Create subset A = {users who watched action content last 30 days} |
| ▼ |
| **STEP 3:** Create subset B = {users who completed at least 3 episodes of a drama} |
| ▼ |
| **STEP 4:** Create subset C = {users who opened the app but watched < 10 min total} |
| ▼ |
| **STEP 5:** Compute A ∩ B = action lovers who also watch drama → show thriller hybrids |
| ▼ |
| **STEP 6:** Compute A ∪ B = any genre-engaged user → show diverse content rows |
| ▼ |
| **STEP 7:** Compute C = 'at-risk' users → show shorter, high-engagement clips first |
| ▼ |
| **STEP 8:** Feed each subset into the appropriate recommendation model variant |

### 3.2.2 Mathematical Formulation

**Content Eligibility — Set Intersection Rule**

$$\text{SHOW}(u, t) = \text{Active}(u) \cap \text{Licensed}(t, \text{region}(u)) \cap \text{AgeOK}(u, \text{rating}(t))$$

A title t is shown to user u only if u is in ALL three sets: active subscribers, the licensed region set, and the age-appropriate set. This Boolean set intersection fires for every (user, title) pair — billions of evaluations per minute across Netflix's infrastructure.

### 3.2.3 Power Set and Feature Combinations

Every OTT recommendation model uses a feature vector for each user. If a user can be characterised by n binary features (has_watched_action, prefers_hindi, is_family_account, etc.), the set of all possible user profiles is the power set P(F) where |F| = n, giving $2^n$ possible profiles. For n = 20 features, this is over 1 million possible user types. Netflix uses dimensionality reduction (PCA, matrix factorisation) to compress this power set into manageable embeddings — but the theoretical framework is the power set.

**Power Set and User Profiles**

```
|P(F)| = 2^n  for n binary features
```

For n=20: 2^20 = 1,048,576 possible user types. Netflix handles this with matrix factorisation — compressing the 2^20 space into ~512-dimensional embedding vectors. This is why your Netflix profile is unique even among 300 million users.

## 3.3 Relations in Banking Systems

### 3.3.1 Binary Relations and Transaction Authorisation

A bank's authorisation system implements a binary relation R ⊆ Accounts × Accounts where (a, b) ∈ R means 'account a is authorised to transfer funds to account b.' This relation is stored in a database table and checked in real-time for every fund transfer. For corporate banking, these relations are complex: a company treasurer account may be authorised to transfer to 50 vendor accounts, but only the CEO account can transfer above a certain threshold.

Properties of this relation matter enormously. The relation is not necessarily symmetric (a → b does not imply b → a). It may be transitive (if a → b and b → c, maybe a can transfer indirectly). Detecting the transitive closure of this relation is used in anti-money laundering (AML) investigations — following the money through chains of transfers.

### 3.3.2 Equivalence Relations — Customer Clustering for Risk

Banks segment customers using equivalence relations. Define a relation R on customer set C: ($c_1$, $c_2$) ∈ R if and only if $c_1$ and $c_2$ have the same credit tier, same income bracket, and same geographic risk zone. If R is an equivalence relation (reflexive, symmetric, transitive), it partitions C into equivalence classes. Each class receives the same credit product offer, the same interest rate, and the same fraud-scrutiny level. This partition is computed by K-means clustering — a machine learning algorithm that is computing equivalence classes on a feature metric space.

**Equivalence Class Partition**

$C = [c_1] \cup [c_2] \cup \ldots \cup [c_k]$ (disjoint classes)

SBI divides its 500M+ customers into risk tiers using ML clustering. Each tier is an equivalence class — customers in the same class get the same interest rate and product recommendations. K-means clustering is literally computing this partition.

### 3.3.3 Partial Orders in Account Product Hierarchy

Banking products are partially ordered by feature richness. A Savings Account ≤ Current Account ≤ Premium Current Account in terms of features and transaction limits. This partial order is not total (Savings and Fixed Deposit are not directly comparable — neither is 'less than' the other in all dimensions). Formally, the product set P with this relation forms a partially ordered set (poset). Banks use this poset to design upgrade flows — when a customer's transaction volume increases, the system suggests the next element in the partial order.

| Syllabus Topic | Banking Application | Specific Example | Real Bank / System |
|---|---|---|---|
| Binary Relation | Transaction authorisation graph | (Account A, Account B) ∈ Authorised | HDFC Corporate Banking |
| Equivalence Relation | Customer segmentation | K-means clusters = equivalence classes | SBI YONO AI Engine |
| Partial Order | Product hierarchy | Savings ≤ Current ≤ Premium | ICICI, Axis Bank product tiers |
| Image of a Set | AML transaction tracing | f(S) = all accounts receiving from S | FIU-India, RBI Compliance |
| Power Set | Feature combination space | 2^n user profiles | All ML-based credit scoring |

## 3.4 Functions in FinTech

### 3.4.1 Bijective Functions — Payment Tokenisation

Every time you tap your phone or card to pay, your 16-digit card number is never transmitted over the network. Instead, a Token — a randomly generated surrogate value — is used. Tokenisation is a function t: CardNumbers → TokenSpace. For the system to work correctly, this function must be:

- Injective (one-to-one): No two cards can share the same token — otherwise, payment could be credited to the wrong account
- Surjective (onto): Every possible token value should be reachable — otherwise, the token space is wasted
- Therefore: t must be a bijection — a perfect one-to-one correspondence

The inverse function $t^{-1}$ is held by the card network (Visa, Mastercard, RuPay) in a Hardware Security Module (HSM) and is used only at settlement time. The mathematical guarantee of bijectivity ensures that settlement is always possible. This bijective tokenisation system protects over 10 trillion USD in annual digital payment value globally.

**UPI Virtual Payment Address (VPA) as a Function**

**`f: PhoneNumber → VPA  (e.g., 9876543210 → user@oksbi)`**

UPI's VPA system is a function mapping phone numbers to virtual addresses. The function is surjective (every VPA is mapped to by some phone) but not injective (users can have multiple phone numbers map to one VPA). The NPCI database implements this function lookup in O(1) using a distributed hash table — itself a mathematical structure based on modular arithmetic (congruence relations).

### 3.4.2 Composite Functions — Multi-Step Verification

UPI payment verification uses composite functions. Let f: DeviceID → SessionToken, g: SessionToken → OTPHash, h: OTPHash → TransactionApproval. The composite function $h \circ g \circ f$: DeviceID → TransactionApproval is what executes when you authenticate a UPI payment. Each function in the composition handles a different security layer: device binding, session management, and OTP verification. The security proof of the overall system requires proving properties of each individual function and how they compose — a standard technique in formal verification.

### 3.4.3 Cantor's Diagonal Argument and Streaming Data

At first glance, Cantor's diagonal argument (syllabus Item 3.12) seems purely theoretical. But it has a surprising FinTech application: proving the impossibility of certain compression schemes for real-time payment data. Cantor proved that the real numbers are uncountably infinite — no bijection exists between natural numbers and real numbers. In FinTech streaming systems (like Stripe's payment event stream), this result implies that you cannot enumerate all possible transaction patterns in advance — there will always be novel fraud patterns that no finite rule set can cover. This theoretical insight motivates the move from rule-based systems to ML-based systems, which can generalise to unseen patterns.

### 3.4.4 Schroeder–Bernstein Theorem in Database Mergers

When Reliance and Disney merged their OTT operations in 2024 ($8.5 billion deal), the combined entity had to merge JioCinema and Hotstar's user databases. The Schroeder–Bernstein theorem guarantees: if there exist injective functions f: A → B and g: B → A, then there exists a bijection between A and B. In practice, this means: if you can map every JioCinema user to a distinct Hotstar equivalent and vice versa, a perfect merged database is theoretically possible. Data engineers use this theorem as a mathematical guarantee before investing in complex merge operations.

### Comparative analysis of machine learning models for the detection of fraudulent banking transactions

Authors: Multiple authors
Journal: Cogent Business & Management (2025)

Key Finding: Using 565,000 real banking transactions, Random Forest achieved 95.79% accuracy for fraud detection — highest among tested models including Neural Networks and Naive Bayes.

# CHAPTER 4: UNIT 2 — COUNTING TECHNIQUES & COMBINATORICS

## 4.1 Why Counting Matters at Scale

Counting problems that seem trivial become profound at industrial scale. How many unique user profiles can Netflix construct from 30 behavioral features? How many routing paths exist in a payment network with 500 bank nodes? How many test configurations must a FinTech startup evaluate before deploying a new fraud model? These questions, answered by combinatorics, directly drive system architecture decisions worth billions of dollars.

**Unit 2 Syllabus Topics Covered**

1. Basic Counting Technique
2. S-Inclusion and Exclusion
3. Pigeon-Hole Principle
4. Permutation and Combination (PnC)
5. Principle of Mathematical Induction
6. The Well Ordering Principle
7. Recursive Definition
8. The Division Algorithm
9. Prime Numbers
10. The Greatest Common Divisor
11. Euclidean Algorithm
12. Fundamental Theorem of Arithmetic

## 4.2 Basic Counting and Inclusion-Exclusion in OTT

### 4.2.1 A/B Testing at Netflix — Multiplication Principle

Netflix runs approximately 1,000 A/B tests per year across its platform. Each test involves splitting users into groups to evaluate different interface variants. When multiple tests run simultaneously, the total number of distinct experiment configurations is the product of the number of variants in each test (multiplication principle — syllabus Item 4.1). Consider a simplified scenario: 3 thumbnail designs × 4 row orderings × 2 autoplay settings × 5 recommendation algorithms = 3 × 4 × 2 × 5 = 120 unique experience configurations. Managing which user sees which configuration requires combinatorial bookkeeping to prevent confounding effects.

**A/B Test Configuration Count**

$$\text{Total configs} = n_1 \times n_2 \times n_3 \times \ldots \times n_k \quad \text{(multiplication principle)}$$

Netflix with 1000 tests of avg 3 variants each: 3^1000 possible global states. In practice, tests are stratified to avoid interactions. Inclusion-exclusion (Syllabus 4.2) is used to count users in multiple test groups simultaneously.

### 4.2.2 Inclusion-Exclusion in Content Licensing

OTT platforms must report the number of users who watched content across different licensing categories for royalty calculations. The Inclusion-Exclusion principle (syllabus Item 4.2) is essential when users watch content from overlapping categories. For example: |Drama ∪ Thriller| = |Drama| +

|Thriller| - |Drama ∩ Thriller|. Without this correction, licensing fees would be double-counted for users who watched both. For k content categories, the full inclusion-exclusion formula involves $2^k$ - 1 terms — computable efficiently using database aggregation.

### 4.2.3 Permutations in Homepage Layout

When Netflix constructs a homepage for a user, it selects and orders content rows. If there are 20 possible row types and the page shows 8 rows, the number of ordered arrangements is the permutation P(20, 8) = 20!/(20-8)! = 5,079,110,400. No human reviews all possible orderings. Netflix's reinforcement learning system (using multi-armed bandit algorithms) efficiently searches this combinatorial space. The bandit algorithm's effectiveness directly depends on understanding the size of the search space — which is the permutation count.

**Permutation Formula Applied**

`P(n,r) = n! / (n-r)!  →  P(20,8) = 5,079,110,400`

Netflix homepage: 5 billion possible orderings per user. ML explores this space using contextual bandits that update policies based on click-through rates. Without combinatorial analysis, engineers couldn't allocate computational resources appropriately.

### 4.2.4 Combinations in Collaborative Filtering

Collaborative filtering — the algorithm behind 'users like you also watched' — measures similarity between pairs of users. For Netflix's 301.6 million subscribers, the number of distinct user pairs is C(301,600,000, 2) = 301,600,000 × 301,599,999 / 2 ≈ $4.5 \times 10^{16}$. This combinatorial explosion makes brute-force pairwise comparison impossible. The insight from combinatorics directly motivates Locality-Sensitive Hashing (LSH) — an approximation algorithm that reduces the effective comparison count from $O(N^2)$ to O(N log N) while preserving similarity structure with high probability.

## 4.3 Pigeonhole Principle in Banking Security

### 4.3.1 UUID Collision Avoidance

Every banking transaction, every database record, every session token requires a unique identifier. If identifiers are generated by a hash function mapping to a 32-bit space (approximately 4.295 billion values), the Pigeonhole Principle (syllabus Item 4.3) guarantees that once more than 4.295 billion transactions exist, at least two will share the same ID — a potentially catastrophic financial error. This is why modern banking systems use UUID v4: 128-bit identifiers with approximately $3.4 \times 10^{38}$ possible values. The probability of a collision across 1 trillion transactions is less than $10^{-15}$ — effectively zero.

### 4.3.2 Birthday Attack and Password Policy

The Birthday Paradox — a consequence of the Pigeonhole Principle — has a direct security implication for banking. In a group of just 23 people, there is a 50% chance two share a birthday. Analogously, in a hash space of size N, a collision can be found with $O(\sqrt{N})$ attempts rather than N attempts. A fraudster using a birthday attack on a 4-digit ATM PIN (10,000 values) needs only $\sqrt{10,000} = 100$ attempts before

expecting a collision. Banks enforce 3-attempt lockout policies explicitly to defeat this counting-based attack.

## 4.4 Mathematical Induction in Algorithm Verification

### 4.4.1 Proving Gradient Descent Convergence

Mathematical induction (syllabus Item 4.5) is the standard tool for proving properties of iterative ML algorithms. Every OTT recommendation model and every banking fraud classifier is trained using gradient descent — an iterative optimisation algorithm. Proving its convergence uses induction:

- Base Case: At training step k=0, the loss function $L_0$ is finite (model is initialised with random small weights)
- Inductive Step: Assume $L_k \leq C$ for some constant C. Under Lipschitz-smooth conditions: $L_{k+1} \leq L_k - \eta||\nabla L||^2 \leq L_k$
- Conclusion: By induction, the loss decreases monotonically at every step, guaranteeing stable training

This inductive argument is not just theoretical — it is the mathematical certificate that justifies deploying gradient descent in production systems. Without it, engineers would have no formal guarantee that training would converge rather than diverge (loss explosion), which would make the model unusable.

### 4.4.2 Recursive Loan EMI Computation

Bank loan EMI systems use recursive definitions (syllabus Item 4.7). The outstanding balance B(n) after n EMI payments satisfies: B(n) = B(n-1) × (1 + r) - P, where r is the monthly rate and P is the fixed EMI. Mathematical induction proves the closed form: $B(n) = B(0)\times(1+r)^n - P\times[(1+r)^n - 1]/r$. This closed form is what every bank's loan management system uses to compute prepayment amounts, foreclosure penalties, and amortisation schedules. The correctness proof is a standard strong induction argument.

**EMI Recursive Formula (Proved by Induction)**

$$B(n) = B(0)\times(1+r)^n - P\times[(1+r)^n - 1]/r$$

Used by every Indian bank for EMI calculation. The Principle of Mathematical Induction (Syllabus 4.5) proves this formula is correct for all n ≥ 1. Without this proof, banks could face legal challenges if their calculations are disputed.

## 4.5 Well Ordering Principle and UPI Routing

The Well Ordering Principle (syllabus Item 4.6) states that every non-empty set of positive integers has a smallest element. This seemingly simple principle has a profound algorithmic consequence: for any finite set of routing costs, an optimal route always exists. Dijkstra's shortest-path algorithm — used by NPCI (National Payments Corporation of India) to route UPI transactions through the banking network — implicitly relies on this principle in its greedy selection step: 'choose the unvisited node with the smallest known distance.'

In 2025, UPI processed 228.3 billion transactions worth ₹299.7 lakh crore. Every one of these transactions was routed through NPCI's network using graph algorithms founded on the Well Ordering Principle. The algorithm runs in O(V log V + E) time (where V = bank nodes, E = interbank connections), enabling sub-second routing decisions at a scale of 21.63 billion transactions per month.

## 4.6 Prime Numbers and FinTech Cryptography

### 4.6.1 RSA Encryption — The Mathematics of Banking Security

Every HTTPS connection to a banking website, every SSL-encrypted API call, every digitally signed UPI transaction uses public-key cryptography — built on the arithmetic of prime numbers (syllabus Items 4.9 and 4.12). RSA (Rivest–Shamir–Adleman) works as follows:

1. Choose two distinct large prime numbers p and q (each 2048 bits long in modern banking)
2. Compute the modulus n = p × q (the public modulus — approximately 617 decimal digits)
3. By the Fundamental Theorem of Arithmetic (syllabus 4.12): n has a unique prime factorisation {p, q}
4. Choose public exponent e and compute private exponent d such that e × d ≡ 1 (mod φ(n))
5. Encrypt message M: C = M^e mod n | Decrypt ciphertext C: M = C^d mod n
6. Security: Finding p and q from n requires factoring a 617-digit number — computationally infeasible

| 2048 | 617 | 10^617 | $10T+ |
|---|---|---|---|
| Min RSA key bits (bank standard since 2015) | Decimal digits in RSA-2048 modulus | Approx number of 2048-bit primes | Annual digital payments secured by RSA/ECC |

### 4.6.2 Euclidean Algorithm in Multi-Currency Settlement

The Euclidean Algorithm (syllabus Item 4.11) computes the Greatest Common Divisor (GCD) in O(log(min(a,b))) steps. In FinTech, it has a direct application in bilateral netting — the process by which banks reduce the number of actual money movements needed to settle mutual obligations. SWIFT's global netting system uses GCD-based algorithms to find the minimum transaction set to clear all obligations. If Bank A owes Bank B $300M and Bank B owes Bank A $200M, the net obligation is $100M (GCD computation), reducing two transactions to one. Globally, this netting saves the banking system an estimated $1 trillion+ in daily liquidity requirements.

**Graph Neural Networks for Financial Fraud Detection: A Review**

Authors: Dawei Cheng et al.

Journal: arXiv:2411.05815 (November 2024)

Key Finding: GNNs are exceptionally adept at capturing complex relational patterns in financial transaction networks, significantly outperforming traditional fraud detection methods. The review covers data types, methods, and real-world deployment scenarios.

# CHAPTER 5: UNIT 3 — PROPOSITIONAL LOGIC & PROOF TECHNIQUES

## 5.1 Logic as the Language of Computing

Propositional logic is not metaphor when applied to computing — it is literal. Every if-else statement is a conditional proposition. Every database WHERE clause is a conjunction of predicates. Every fraud rule is a Boolean formula. Every access control policy is a system of implications. The connectives AND (∧), OR (∨), NOT (¬), IMPLIES (→), and IFF (↔) are the operators that software engineers write thousands of times per day, often without realising they are doing formal logic.

**Unit 3 Syllabus Topics Covered**

PROPOSITIONAL LOGIC:

1. Basic Connectives and Truth Tables
2. Logical Equivalence — Laws of Logic
3. Logical Implication
4. Rules of Inference
5. The Use of Quantifiers

PROOF TECHNIQUES:

6. Some Terminology
7. Proof Methods and Strategies
8. Forward Proof
9. Proof by Contradiction
10. Proof by Contrapositive
11. Proof of Necessity and Sufficiency

## 5.2 Propositional Logic in OTT Platforms

### 5.2.1 Content Eligibility as a Truth Table

Every OTT platform must evaluate whether a specific title should be shown to a specific user. This eligibility check is a propositional formula evaluated across millions of (user, title) pairs every second. The formula for Netflix content display can be formalised as:

**Netflix Content Eligibility Formula**

```
SHOW(u,t) ≡ (active(u) ∧ licensed(t, region(u)) ∧ ¬restricted(u, rating(t)))
```

This is a conjunction of three propositions. A truth table of size $2^3 = 8$ covers all cases. Netflix evaluates this for 301.6M users × 15,000+ titles = 4.5 trillion evaluations — optimized using pre-computed indexes built on De Morgan's Law simplifications.

| active(u) | licensed(t) | ¬restricted(u,t) | SHOW(u,t) | Action |
|---|---|---|---|---|
| T | T | T | TRUE | Show content |
| T | T | F | FALSE | Block (age restriction) |
| T | F | T | FALSE | Block (not licensed in region) |
| T | F | F | FALSE | Block (both reasons) |
| F | T | T | FALSE | Block (subscription expired) |
| F | T | F | FALSE | Block (subscription + age) |

| active(u) | licensed(t) | ¬restricted(u,t) | SHOW(u,t) | Action |
|---|---|---|---|---|
| F | F | T | FALSE | Block (subscription + region) |
| F | F | F | FALSE | Block (all reasons) |

### 5.2.2 De Morgan's Law in Database Query Optimisation

De Morgan's Laws (Logical Equivalence — syllabus Item 5.2) are used constantly in database query optimisation. Suppose Netflix's recommendation filter must exclude banned users AND users in restricted regions: ¬banned(u) ∧ ¬restricted(u). By De Morgan: ¬banned(u) ∧ ¬restricted(u) ≡ ¬(banned(u) ∨ restricted(u)). The second form allows the query optimiser to use a single composite index on the OR condition, reducing the query from two full table scans to one indexed lookup — critical when checking conditions across 300 million user records.

## 5.3 Rules of Inference in Banking Fraud Detection

### 5.3.1 Modus Ponens — The Foundation of Rule Engines

Banking fraud detection rule engines are, at their mathematical core, Modus Ponens (MP) machines. Modus Ponens states: given p and p → q, conclude q. A bank's fraud detection engine contains hundreds of rules of the form p → q (if condition, then action). When all premises are simultaneously true for a transaction, the conclusion fires in milliseconds.

**Live Fraud Rule Engine Example (HDFC Bank)**

RULE 47: IF (txn_amount > ₹50,000) AND (time = night: 11PM-4AM) AND (device = new_device) AND (location = international) THEN FRAUD_RISK = CRITICAL

Logical form: $p_1 \wedge p_2 \wedge p_3 \wedge p_4 \rightarrow q$

$p_1$: amount > 50000 [TRUE for this transaction]
$p_2$: time in {23:00–04:00} [TRUE]
$p_3$: device_id not in known_devices [TRUE]
$p_4$: IP geolocation != India [TRUE]
→ q: FRAUD_RISK = CRITICAL [By Modus Ponens: FIRES]

Result: Transaction blocked; customer gets SMS + call within 30 seconds

### 5.3.2 Modus Tollens — Login Anomaly Detection

Modus Tollens (MT) states: given ¬q and p → q, conclude ¬p. Bank login systems use MT for geolocation anomaly detection. The rule: 'If user is genuinely in India (p), then their IP is an Indian IP (q).' If the system detects a non-Indian IP (¬q), it concludes by MT that the user is not genuinely in India (¬p) — triggering step-up authentication (OTP, biometric re-verification). This logical inference runs on every login event at SBI YONO, HDFC NetBanking, and every major Indian bank's digital platform.

### 5.3.3 Hypothetical Syllogism in Transaction Chains

Hypothetical Syllogism (HS): from p → q and q → r, conclude p → r. In banking fraud investigation, this allows chaining of implication rules to reach conclusions that no single rule can reach. Example: Rule 1: 'If account is newly opened (p), then transactions above ₹10,000 should be flagged (q).' Rule 2: 'If a flagged transaction occurs at 3AM (q and r), then freeze account pending review (s).' By HS: if account is newly opened and a ₹10,000+ transaction occurs at 3AM, freeze the account. This chain fires automatically across India's banking fraud systems.

## 5.4 Quantifiers in Database Queries

### 5.4.1 Universal Quantifier (∀) in AML Reporting

The Reserve Bank of India requires banks to report customers whose ALL transactions in a given period exceed certain thresholds. This is a universal quantification: ∀t ∈ Transactions(customer, 30 days): t.amount > ₹1,00,000. In SQL: SELECT customer_id FROM transactions WHERE transaction_date > NOW()-30 GROUP BY customer_id HAVING MIN(amount) > 100000. The SQL HAVING MIN clause is exactly the universal quantifier — flagging only those customers for whom every transaction exceeds the threshold. This query runs daily across all Indian bank databases for RBI Suspicious Transaction Reports.

### 5.4.2 Existential Quantifier (∃) in Fraud Pattern Detection

Existential quantification (∃) maps to SQL EXISTS. Example: Flag any account that has AT LEAST ONE transaction with a new merchant AND a transaction amount > 5× the account's average. Formally: ∃t ∈ Transactions(account): new_merchant(t) ∧ (t.amount > 5 × avg_amount(account)). This SQL query, using EXISTS and a correlated subquery, is the existential quantifier made operational. Banks run hundreds of such queries daily as part of their automated transaction monitoring systems.

## 5.5 Proof Techniques in System Design

### 5.5.1 Direct Proof — RSA Correctness

The correctness of RSA decryption — that decrypting the encrypted message always recovers the original — is proved directly using Fermat's Little Theorem and the Chinese Remainder Theorem. The proof is: for RSA parameters (n, e, d), for any message M: M^(ed) ≡ M (mod n). This is proved directly using Euler's theorem: if gcd(M,n) = 1, then M^φ(n) ≡ 1 (mod n). Since ed ≡ 1 (mod φ(n)), we have ed = kφ(n)+1, and M^(ed) = M^(kφ(n)+1) = M × (M^φ(n))^k ≡ M × 1^k = M (mod n). Every bank's SSL/TLS certificate relies on this direct proof being correct.

### 5.5.2 Proof by Contradiction — Impossibility of Perfect Security

Proof by contradiction (syllabus Item 5.9) is used to establish fundamental impossibility results. In cryptography: Assume (for contradiction) that a system exists that is simultaneously unbreakable, instantaneous, and zero-cost. An unbreakable system must use one-time pads (proved by Shannon,

1949 — also a proof by contradiction). But one-time pads require key lengths equal to message lengths — contradicting 'zero-cost' for long messages, or require secure key distribution — contradicting 'zero additional infrastructure.' Therefore, no such system exists. This proof guides every FinTech company's security budget allocation.

### 5.5.3 Necessary and Sufficient Conditions in Loan Approval

Distinguishing necessary from sufficient conditions (syllabus Item 5.11) prevents critical logical errors in AI system design. In loan approval: A CIBIL score ≥ 750 is a NECESSARY condition for approval at most Indian banks (you will not get a loan without it), but NOT a SUFFICIENT condition (other factors like income, employment stability, debt-to-income ratio are also required). An ML model that learns only sufficiency (if all conditions met → approve) without encoding the necessary condition (if CIBIL < 750 → always reject) will make approval errors that increase Non-Performing Assets (NPAs). India's banking sector carries ₹2.09 lakh crore in NPAs — a significant portion attributable to faulty credit decision logic.

| | | | |
|---|---|---|---|
| **₹2.09L cr** | **750+** | **99.99%** | **<50ms** |
| Banking NPAs India FY2024 | CIBIL score needed (necessary condition) | Bank uptime SLA (logical guarantee needed) | Max fraud decision latency (UPI standard) |

## 5.6 Generative AI and Logic — LLM Guardrails in Banking

Large Language Models (LLMs) like GPT-4, Claude, and Gemini are now deployed in banking customer service chatbots (HDFC Eva, Axis Aha, SBI SIA) and FinTech applications. These models generate natural language, but in financial contexts, their outputs must be logically constrained. This is done using Constrained Decoding — a system that applies propositional logic rules to LLM output tokens in real-time:

- Rule: IF (output contains 'interest rate' mention) → THEN rate must be in [0%, 36%]
- Rule: IF (output contains 'guaranteed returns') → THEN flag for human review (SEBI compliance)
- Rule: IF (output contains personal account details) → THEN suppress output immediately
- Rule: IF (user is asking about loan eligibility) → THEN always include CIBIL requirement
- These rules are propositional formulas evaluated on every token generated by the LLM

These guardrails are built using the same propositional logic tools studied in Unit 3. The entire field of AI Safety — which includes ensuring LLMs deployed in banking give correct, legal, and non-harmful financial advice — is built on formal logic foundations.

**Enhancing Fraud Detection in Banking With Deep Learning: GNNs and Autoencoders**

Authors: Alarfaj FK, Shahzadi S.

Journal: IEEE Access (2025)

Key Finding: GNNs and autoencoders effectively detect fraud with a balanced precision-recall tradeoff, improving the efficiency of banking systems while enabling real-time credit card fraud prevention.

# CHAPTER 6: UNIT 4 — ALGEBRAIC STRUCTURES & MORPHISM

## 6.1 From Abstract Algebra to Real Industry

Algebraic structures — groups, rings, fields, monoids, Boolean algebras — may be the most abstract chapter in the discrete mathematics syllabus. They are also the most secretly ubiquitous in industry. The reason: abstract algebra provides properties that hold universally across all structures of a given type. Once you prove a result for any group, it applies to integers under addition, to permutations of API tokens, to points on an elliptic curve, to the set of symmetric encryption keys under composition. This generality is precisely why cryptography, the mathematical foundation of all banking and FinTech security, is built on algebraic structures.

**Unit 4 Syllabus Topics Covered**

1. Algebraic Structures with One Binary Operation
2. Semi Groups 3. Monoids 4. Groups
5. Congruence Relations and Quotient Structures 6. Free and Cyclic Monoids and Groups
7. Permutation Groups 8. Substructures 9. Normal Subgroups
10. Algebraic Structures with Two Binary Operations 11. Rings
12. Integral Domain and Fields 13. Boolean Algebra and Boolean Ring
14. Identities of Boolean Algebra 15. Representation of Boolean Functions
16. Disjunctive and Conjunctive Normal Form

## 6.2 Groups in Cryptography

### 6.2.1 Cyclic Groups and RSA

RSA encryption, which secures every banking website and every TLS handshake, is built on the multiplicative group of integers modulo n: $(\mathbb{Z}/n\mathbb{Z})^*$, where n = p × q (product of two large primes). This is a cyclic group of order φ(n) = (p-1)(q-1). The RSA encryption operation M^e mod n is a group operation — raising an element to a power in the group. Decryption M^d mod n is the inverse operation. The security relies on the computational hardness of the discrete logarithm problem in this group: given M^e mod n, find e without knowing the group's factorisation.

### 6.2.2 Elliptic Curve Cryptography (ECC) — Banking's Modern Standard

Modern banking has largely moved from RSA to Elliptic Curve Cryptography (ECC) for digital signatures. ECC is based on the group of points on an elliptic curve over a finite field. The group law defines 'point addition': given two points P and Q on the curve, their 'sum' P ⊕ Q is a third specific point on the curve. This operation is associative, has an identity element (the 'point at infinity'), and every element has an inverse — making the point set an abelian group under ⊕.

| RSA | ECC (Modern Standard) |
|---|---|
| RSA-2048: ~256 bytes key | ECC-256: ~32 bytes key |
| Key generation: slow (~minutes on IoT) | Key generation: 10x faster than RSA |
| Standard since 1977 | Standard since 2000s |
| Quantum vulnerable (Shor's algorithm) | Also quantum vulnerable (modified Shor's) |
| Used in older TLS systems | Used in Apple Pay, Google Pay, UPI |

UPI uses ECC-based digital signatures to authenticate every transaction. When you confirm a UPI payment, your phone signs the transaction data using a private key stored in the Trusted Execution Environment (TEE). The bank verifies the signature using the corresponding public key. This entire process — sign, transmit, verify — executes in under 50 milliseconds, enabling UPI's sub-second payment confirmation even at 21 billion transactions per month.

### 6.2.3 Permutation Groups in API Token Rotation

Banking Open Banking APIs (mandated by RBI under the Account Aggregator framework) use OAuth 2.0 token management. Session tokens are structured as permutations of base strings — drawing on the symmetric group $S_n$ (syllabus Item 4.7: Permutation Groups). Valid tokens at any time correspond to a specific subgroup $H \leq S_n$. Token validation is group membership testing: does this token belong to H? This is O(1) using group representation, compared to O(N) for blacklist-based revocation. As HDFC Bank alone has millions of active API sessions, this group-theoretic efficiency is practically critical.

## 6.3 Rings and Fields in Machine Learning

### 6.3.1 The Ring Structure of Neural Networks

Every deep learning model — whether for OTT recommendations, banking fraud detection, or FinTech credit scoring — performs its core computation using matrix multiplication. The set of n×n matrices over real numbers, $(M_n(\mathbb{R}), +, \times)$, forms a non-commutative ring (syllabus Item 4.11): matrix addition is commutative, matrix multiplication is not. The weighted sum computation W×x + b (where W is a weight matrix, x is input, b is bias) is a ring operation. Every forward pass of a neural network is a sequence of ring operations, performed trillions of times per day across data centres serving these three industries.

**Neural Network Layer Computation (Ring Operation)**

$$y = \sigma(Wx + b) \quad \text{where } (M_n(\mathbb{R}), +, \times) \text{ forms a non-commutative ring}$$

Netflix runs inference for 300M users across thousands of GPU servers. Each inference is thousands of matrix ring operations. The ring structure (closure, associativity, distributivity) is what makes GPU parallelism possible — matrix multiplication can be distributed across cores because ring operations are compatible with parallelism.

### 6.3.2 Integral Domains in FinTech Exact Arithmetic

Financial calculations demand exact arithmetic — no floating-point rounding errors. The ring of integers $\mathbb{Z}$ is an integral domain (syllabus Item 4.12): no zero divisors. If $ab = 0$ in $\mathbb{Z}$, then $a = 0$ or $b = 0$. This means integer arithmetic has no 'accidental zeros' from rounding. FinTech systems store all monetary amounts in the smallest currency unit (paisa for INR, cents for USD, satoshi for Bitcoin) as integers. All arithmetic is performed in $\mathbb{Z}$ and only converted to decimal for display. This practice, grounded in the integral domain property of $\mathbb{Z}$, prevents the systematic rounding errors that once caused banks to accumulate fractions of currency across millions of transactions.

### 6.3.3 Boolean Algebra in Fraud Detection Rule Engines

Boolean Algebra (syllabus Items 4.13–4.16) is the most directly visible algebraic structure in banking software. A production fraud rule engine at any Indian bank evaluates expressions of the form:

**Fraud Detection Boolean Expression**

$$\mathbf{FRAUD = (amt > thresh) \wedge \neg(whitelist) \wedge (vel > 3) \vee (new_dev \wedge intl)}$$

Using Boolean algebra identities: Distributive Law lets the engine factor common subexpressions. Absorption Law eliminates redundant checks. De Morgan's reduces NOT gates. The simplified form executes faster on the bank's transaction processing hardware — critical at Visa's 65,000 transactions/second.

### 6.3.4 Karnaugh Maps and ASIC Optimisation

Payment terminal chips (used in every POS machine, ATM card reader, and contactless payment device) are custom ASICs (Application-Specific Integrated Circuits) designed for maximum speed at minimum power. The logic circuits inside them are designed using Karnaugh Maps (K-maps) — a graphical method for Boolean function minimisation that is directly based on the Disjunctive Normal Form (DNF) and Conjunctive Normal Form (CNF) concepts from syllabus Item 4.16. By grouping 1-cells in the K-map, engineers find the minimal Sum of Products (SOP) expression, reducing the number of logic gates needed. Fewer gates means smaller chip area, lower power consumption, and faster operation — directly translating to the energy efficiency of the 100+ million POS terminals deployed globally.

## 6.4 Monoids and Semigroups in Data Processing

### 6.4.1 Monoids in Transaction Batching (FinTech)

A monoid is a set with an associative binary operation and an identity element. The set of all valid payment instruction sequences under concatenation forms a free monoid (syllabus Item 4.6): the

operation is associative (concatenating sequences), the identity is the empty sequence, and any combination of valid instructions produces a valid sequence. FinTech payment processing pipelines use this monoid structure to safely parallelize batch processing: since concatenation is associative, a batch of 1 million transactions can be split into 1,000 groups of 1,000, processed independently, and combined — with the result identical regardless of split strategy.

### 6.4.2 Homomorphic Encryption — The Future of Privacy-Preserving FinTech

Homomorphic Encryption (HE) is arguably the most exciting application of ring homomorphisms (algebraic morphisms — syllabus Chapter 4 theme) in FinTech today. HE allows computation on encrypted data without decryption: Encrypt(a + b) = Encrypt(a) $\oplus$ Encrypt(b), where $\oplus$ is the 'encrypted addition' in the ciphertext ring. This is exactly a ring homomorphism: the encryption function $\varphi$: (plaintext ring, +) $\rightarrow$ (ciphertext ring, $\oplus$) preserves the ring structure.

The banking implication is revolutionary: a bank can run a fraud detection ML model on fully encrypted customer transaction data — without ever seeing the actual amounts, merchants, or account numbers. The model outputs an encrypted fraud score, which only the customer can decrypt with their private key. Microsoft Azure and Google Cloud now offer Homomorphic Encryption as a cloud service. Indian FinTech startups are beginning to deploy HE for privacy-preserving credit scoring, where credit bureaus can compute scores on encrypted financial data without accessing raw records.

**Financial fraud detection using a hybrid deep belief network and quantum optimization approach**

Authors: Multiple authors
Journal: Discover Applied Sciences / Springer Nature (2025)

Key Finding: A novel hybrid model integrating Deep Belief Networks with quantum optimization, CNNs, LSTMs, and GNNs achieves improved accuracy, training speed, and robustness in financial fraud detection — demonstrating that algebraic structures (ring operations in neural networks) enable this multi-modal approach.

## 6.5 Congruence Relations in Distributed Systems

Congruence relations (syllabus Item 4.5) power consistent hashing — the standard technique for distributing data across server clusters. In a consistent hash ring, server IDs and data keys are both mapped to positions on a circle via h(x) = x mod N (where N is the ring size). This is congruence modulo N. Two keys are congruent if they hash to the same server. The congruence class structure ensures that when a server is added or removed, only O(1/N) fraction of keys need to be remapped — far more efficient than naive modular hashing. Netflix uses consistent hashing to distribute content metadata across its distributed Cassandra database clusters; NPCI uses it to distribute UPI transaction records across its data centres.

| Algebraic Structure | Syllabus Item | Industry | Application | Key Property Used |
|---|---|---|---|---|
| Cyclic Group | 4.4 Groups | Banking | RSA encryption | Discrete logarithm hardness |
| Abelian Group | 4.4 Groups | FinTech (UPI) | ECC digital signatures | Group law on elliptic curve |
| Permutation Group | 4.7 Perm. Groups | Banking APIs | OAuth token rotation | Group membership testing |
| Non-comm. Ring | 4.11 Rings | OTT/Banking ML | Neural network computation | Matrix multiplication (ring ×) |
| Integral Domain | 4.12 Int. Domain | FinTech | Exact currency arithmetic | No zero divisors (no rounding) |
| Boolean Algebra | 4.13-14 | Banking | Fraud rule engine | Complement, AND, OR, De Morgan |
| DNF/CNF | 4.16 | FinTech Hardware | ASIC/K-map optimisation | SOP/POS minimisation |
| Free Monoid | 4.6 Monoids | FinTech | Payment batch processing | Associativity enables parallelism |
| Ring Homomorphism | 4.11 + morphism | FinTech | Homomorphic encryption | Structure-preserving computation |
| Congruence Relation | 4.5 Congruence | OTT/FinTech | Consistent hashing | Modular partitioning |

# CHAPTER 7: UNIT 5 — GRAPHS AND TREES

## 7.1 The World is a Graph

Graph theory is the most visually intuitive and industrially impactful chapter of discrete mathematics. The internet is a graph. A social network is a graph. A supply chain is a graph. A financial transaction network is a graph. A knowledge base used by an AI assistant is a knowledge graph. Every time you use GPS navigation, send a WhatsApp message, watch a Netflix series, or make a UPI payment, graph algorithms are executing somewhere in the pipeline, routing information through millions of connected nodes.

In this chapter, we examine every Unit 5 topic through three industrial lenses, backed by published research, real company data, and worked algorithmic examples. We will see that the Netflix recommendation engine, the UPI payment routing system, the banking fraud detection network, the CBDC blockchain ledger, and the SBI ATM replenishment logistics system are all, at their mathematical core, graph problems.

**Unit 5 Syllabus Topics Covered**

1. Graph and their properties  2. Degree  3. Connectivity  4. Path and Cycle
5. Sub Graph  6. Isomorphism  7. Eulerian and Hamiltonian Walks
8. Graph Colouring  9. Colouring Maps  10. Planar Graphs
11. Four Colour Conjecture  12. Trees and Rooted Trees  13. Binary Tree

## 7.2 Graph Properties in OTT Recommendation Systems

### 7.2.1 Netflix Knowledge Graph — Scale and Structure

Netflix maintains a proprietary knowledge graph — a labeled, directed multigraph G = (V, E, L) where V represents entities (users, titles, actors, directors, genres, themes, languages, awards), E represents relationships between entities, and L assigns typed labels to edges. The scale of this graph is extraordinary:

- Node types: Users (301.6M), Titles (15,000+), Actors (500,000+), Directors (50,000+), Genres (100+), Awards (1,000+)
- Edge types: watched, rated, acted_in, directed_by, belongs_to_genre, won_award, similar_to, searched_for
- Scale: Hundreds of millions of nodes, billions of edges
- Update rate: Real-time — billions of new 'watched' and 'rated' edges added daily
- Technology stack: Apache Cassandra (distributed storage) + custom graph query engine
- Application: Powers 'More Like This', 'Because you watched', actor filmography rows

Graph Neural Networks (GNNs) — machine learning models that operate directly on graph structures — are now Netflix's primary tool for recommendation. A GNN performs 'message passing': each node aggregates information from its neighbours, allowing the model to learn that a user who watched

Christopher Nolan films is likely to enjoy Denis Villeneuve films (both are connected through genre nodes, award nodes, and collaborative filtering edges).

### 7.2.2 Degree Centrality and Popularity Scoring

The degree of a node in a graph (syllabus Item 7.2) measures how many edges are incident to it. In the Netflix content graph, the in-degree of a title node — the number of 'watched' edges pointing to it from user nodes — directly measures viewing popularity. However, raw in-degree is biased toward older content (more time to accumulate views). Netflix normalises by days-since-release, computing adjusted degree centrality:

**Adjusted Degree Centrality for Content**

```
score(t) = deg_in(t) / sqrt(days_since_release(t) + 1)
```

This formula surfaces content that is gaining traction quickly. A title released 7 days ago with 10M views scores higher than a title released 3 years ago with 100M total views. This 'velocity-adjusted degree centrality' determines which titles appear in the Trending Now row.

### 7.2.3 Graph Connectivity and Recommendation Coverage

A connected graph has a path between every pair of vertices (syllabus Item 7.3). Netflix monitors the connectivity structure of its recommendation graph to ensure every title has a path to at least one user node. A title with no connected users — zero engagement since release — is in an isolated component of the content graph. Netflix's recommendation team flags any title with zero recommendations within 30 days of release as a potential recommendation failure, triggering promotional interventions. This is literally checking graph connectivity.

## 7.3 Paths, Cycles, and Walks in Banking

### 7.3.1 Transaction Graph and Fraud Path Detection

A bank's transaction graph G = (V, E, w) has accounts as vertices and transactions as directed edges with weights representing transaction amounts and timestamps. Fraud rings — groups of coordinated accounts working together to commit fraud — form dense subgraphs within this sparse overall graph. Detecting them requires path analysis: finding all paths from flagged source accounts to destination accounts within k hops.

BFS (Breadth-First Search) on the transaction graph gives all accounts reachable from a suspicious account within k transactions. For k=3, BFS explores: depth-1 (direct transaction targets), depth-2 (recipients of those targets), depth-3 (final destination of laundered funds). A 2024 study in the Journal of Safety Science and Resilience demonstrated that dynamic heterogeneous transaction graph embedding (DyHDGE) achieves state-of-the-art fraud detection by exploiting temporal graph structure — the time ordering of edges in the transaction graph.

| Banking Fraud Ring Detection Using Graph BFS |
|---|
| **STEP 1:** Build transaction graph G: accounts=nodes, transactions=directed edges |
| ▼ |
| **STEP 2:** Identify seed set S: accounts flagged by rule engine or reported as suspicious |
| ▼ |
| **STEP 3:** Run BFS from each s ∈ S: collect all accounts within depth k=3 |
| ▼ |
| **STEP 4:** Compute subgraph G[S ∪ BFS(S)]: extract the suspicious subgraph |
| ▼ |
| **STEP 5:** Apply GNN model on subgraph: classify each node as fraud / not fraud |
| ▼ |
| **STEP 6:** Flag high-probability fraud nodes: alert compliance team for investigation |
| ▼ |
| **STEP 7:** Outcome: Fraud rings detected that no single transaction rule could catch |

### 7.3.2 Eulerian Circuits in Compliance Audit Design

An Eulerian circuit traverses every edge of a graph exactly once and returns to the starting vertex (syllabus Item 7.7). A necessary and sufficient condition is that all vertices have even degree (Euler's theorem, 1736). In banking compliance, annual audits must examine every transaction category exactly once without redundancy. Designing the audit sequence as an Eulerian circuit on the transaction-category graph ensures completeness (every type examined) and efficiency (no type examined twice). Euler's theorem is verified programmatically by the audit scheduling system before the compliance calendar is published.

### 7.3.3 Hamiltonian Paths — ATM Cash Logistics

The State Bank of India operates over 68,000 ATMs across India — the world's largest ATM network by count. Each ATM must be refilled with cash periodically. A cash van must visit a subset of ATMs in a day, minimising total travel distance. This is the Hamiltonian Path Problem (HPP) — finding a path that visits every vertex exactly once (syllabus Item 7.7). Unlike the Eulerian circuit, HPP is NP-complete (no polynomial-time algorithm is known). Banks use heuristic algorithms:

- Nearest Neighbour Heuristic: Always visit the closest unvisited ATM next
- 2-OPT Local Search: Iteratively swap path segments to reduce total distance
- Genetic Algorithm: Evolve a population of routes, selecting fitter offspring
- Result: Routes within 5-15% of optimal, saving crores in fuel and logistics costs per year

## 7.4 Graph Colouring in FinTech and Banking

### 7.4.1 Register Allocation in Payment Processors

Graph colouring (syllabus Item 7.8) is one of the most important applications of graph theory in systems programming. Visa's payment processing software runs at 65,000 transactions per second globally. The compiled machine code must execute with maximum efficiency. During compilation, the register allocation problem is modelled as a graph colouring problem: variables that are simultaneously 'live' in memory are connected by an edge in the interference graph, and CPU registers are the colours. Minimising the chromatic number of this interference graph minimises memory spills — the performance bottleneck in high-frequency financial software.

### 7.4.2 Time-Slot Scheduling for RBI Regulatory Reporting

Indian banks must submit multiple regulatory reports to the Reserve Bank of India daily, weekly, and monthly. Some reports share data dependencies — they cannot be generated simultaneously because both read from the same database table, causing locking conflicts. Modelling report types as graph vertices and data conflicts as edges, the minimum number of non-conflicting time slots needed is exactly the chromatic number $\chi(G)$ of this conflict graph. Banks compute $\chi(G)$ using greedy graph colouring algorithms to produce an optimal reporting schedule.

**Graph Chromatic Number and Scheduling**

**$\chi(G)$ = minimum number of time slots = minimum colours for conflict graph G**

A bank with 50 report types and 30 conflict edges might have $\chi(G) = 5$, meaning only 5 time slots are needed — vs. 50 if run sequentially. This is computed using greedy colouring. Saves hours of batch window time daily across India's 200+ scheduled commercial banks.

### 7.4.3 The Four Colour Theorem in Geographic Risk Mapping

The Four Colour Theorem (syllabus Item 7.11) states that any planar graph (representing a map) can be coloured with at most 4 colours such that no adjacent regions share a colour. Banks and insurance companies apply this directly to geographic risk zone maps: assign risk levels (Low, Guarded, High, Critical — 4 levels) to geographic regions such that no two bordering regions have the same risk level. This prevents risk map confusion for field agents and loan officers. The theorem guarantees that 4 risk levels are always sufficient for any geographic map — no matter how complex.

## 7.5 Trees and Their Applications

### 7.5.1 Decision Trees in Credit Scoring and Fraud Detection

Decision trees (syllabus Item 7.12 — Trees and Rooted Trees) are a primary machine learning technique named for their tree structure: a rooted tree where internal nodes test features, branches represent outcomes, and leaves assign class labels. They are the most interpretable ML model — unlike neural networks ('black boxes'), a decision tree's reasoning can be read like a flowchart, which

is essential for regulatory compliance in banking (banks must be able to explain their lending decisions under RBI's Fair Practices Code).

**Decision Tree: HDFC Bank Home Loan Approval Path**

ROOT: CIBIL Score ≥ 750?
  └─ YES: Monthly Income ≥ ₹40,000?
    └─ YES: Existing EMI burden < 40% of income?
      └─ YES: Employment type?
        ├─ SALARIED (≥2 years): APPROVE at 8.40% p.a.
        └─ SELF-EMPLOYED: Additional ITR verification required
      └─ NO: REJECT (over-leveraged)
    └─ NO: REJECT (insufficient income)
  └─ NO: REJECT (CIBIL requirement not met — necessary condition)

Random Forest = 500 such trees; majority vote determines final decision
Model accuracy: 95.79% (Cogent Business & Management, 2025)

### 7.5.2 Binary Search Trees and Database Indexing

Binary Search Trees (BSTs — syllabus Item 7.13) are the conceptual foundation for database B+ tree indexes — the data structure that makes account lookup fast in banking systems. A B+ tree with branching factor k has height O(log_k N) for N records. SBI's 500 million accounts, indexed with k=1000, require only $O(\log_{1000}(500{,}000{,}000)) = O(3)$ node accesses per query — enabling sub-millisecond account retrieval during card swipes, even at 15+ billion daily UPI transactions.

**B+ Tree Height Formula**

$$\text{height}(T) = \lceil \log_k(N) \rceil \quad \rightarrow \quad \log_{1000}(500{,}000{,}000) = \lceil 3 \rceil = 3 \text{ lookups}$$

SBI's database of 500M accounts, indexed with B+ tree (k=1000), needs only 3 disk reads to find any account. This is why ATM card validation completes in ~100ms even for India's largest bank. The BST ordering property (left child < parent < right child) is directly used to bound search depth.

### 7.5.3 Merkle Trees in Blockchain and Digital Currency

Merkle trees — binary trees where each leaf stores the hash of a data block, and each internal node stores the hash of its children (syllabus Item 7.13: Binary Tree) — are the structural foundation of blockchain technology. Every block in Bitcoin, Ethereum, and India's digital rupee (RBI CBDC pilot) contains a Merkle root — the hash at the root of the Merkle tree of all transactions in that block. The key properties:

- Tamper-evidence: Changing any leaf (transaction) changes all hashes up to the root — detectable immediately
- Efficient membership proof: Proving a transaction is in a block requires only O(log N) hashes (tree height)
- Efficient comparison: Two blocks with the same Merkle root contain identical transaction sets — O(1) comparison

- Scalability: A block with 10,000 transactions needs only $\log_2(10{,}000) \approx 14$ hash comparisons for membership proof

The RBI's CBDC (Central Bank Digital Currency — 'Digital Rupee') pilot uses Merkle tree-based transaction verification. UPI's potential integration with blockchain settlement is also based on this structure. The FinTech blockchain market, valued at USD 9.95 billion in 2024 and projected at USD 46.87 billion by 2030 (CAGR 46.92%), is entirely built on this binary tree structure.

### 7.5.4 Spanning Trees in Netflix CDN Topology

Netflix's Open Connect CDN (Content Delivery Network) delivers video to 300+ million users through a global network of servers, data centres, and internet exchange points. This network is modelled as a weighted graph G = (V, E, w) where V are servers/data centres, E are fibre/network links, and w(e) is the link cost (bandwidth cost, latency, or installation cost). The Minimum Spanning Tree (MST) of this graph, computed using Kruskal's or Prim's algorithm, gives the optimal backbone topology: connecting all nodes with minimum total link cost. Netflix uses the MST as the base topology and then adds back selected redundant edges for fault tolerance.

**Kruskal's MST Algorithm Complexity**

**Time: O(E log E) → For |E|=10,000 CDN links: ~130,000 operations**

Netflix's CDN has thousands of nodes globally. MST computation takes milliseconds on modern hardware. The MST identifies which server-to-server links are essential for connectivity, guiding billion-dollar infrastructure investment decisions. This is a direct application of Syllabus 7.12 (Trees).

### 7.5.5 Rooted Trees in Organisational Hierarchies (Banking)

Bank organisational charts are rooted trees. The root is the Board / CEO. Internal nodes are divisions and departments. Leaves are individual branches and staff positions. This tree structure determines the access control hierarchy: a branch manager (leaf-level node) can approve loans up to ₹25 lakh; a regional manager (one level up) can approve up to ₹5 crore; only the national credit committee (near the root) can approve above ₹100 crore. Banking software implements these approval workflows using tree traversal algorithms — specifically, the tree's depth determines the approval escalation path.

## 7.6 Sub-Graphs and Community Detection in Fraud

### 7.6.1 Dense Sub-Graphs as Fraud Rings

Financial fraud rings are organised groups of people who coordinate to commit fraud — creating fake transactions, money mule networks, or synthetic identity fraud. In a transaction graph, these rings appear as unusually dense sub-graphs (syllabus Item 7.5): clusters of vertices with many more internal edges than external edges. Community detection algorithms identify these dense sub-graphs:

- Louvain Modularity: Maximises modularity Q = (edges within community)/(total edges) - (expected edges by random model)

- Label Propagation: Each node adopts the majority label of its neighbours — converges to communities
- Graph Convolutional Networks: Learn community structure by aggregating neighbour features
- Result: A 2024 NVIDIA research report shows GNN-based fraud detection outperforms rule-based systems by 5-6x

**Graph Learning-Empowered Financial Fraud Detection: Progress and Future Directions**

Authors: Li Y et al.
Journal: Intelligent Computing / Science AAAS (2025)

Key Finding: Graph learning leveraging relational structure considerably enhances financial fraud detection. Semi-GNN addresses the challenge of mostly unlabeled fraud data by combining supervised and unsupervised learning on transaction graphs.

## 7.7 Isomorphism in Open Banking API Design

Two graphs $G_1$ and $G_2$ are isomorphic (syllabus Item 7.6) if there exists a bijection between their vertex sets that preserves edge structure. In Open Banking (mandated by RBI's Account Aggregator framework), different financial institutions must exchange data using compatible schemas. These schemas can be modelled as graphs where data fields are vertices and dependencies between fields are edges. Automated schema matching tools check for graph isomorphism (or near-isomorphism using subgraph matching) between two institutions' API schemas, identifying compatible fields and flagging incompatibilities. This graph-theoretic approach reduces manual API integration effort by an estimated 70%.

## 7.8 Planar Graphs in Hardware Design

A planar graph can be drawn in a plane without edge crossings. Kuratowski's theorem states a graph is planar if and only if it contains no subdivision of $K_5$ (complete graph on 5 vertices) or $K_{3,3}$ (complete bipartite graph). Planar graphs are essential in Printed Circuit Board (PCB) design for payment terminals, ATM control boards, and contactless card readers. If the circuit's component connection graph is non-planar, the PCB requires multiple layers — increasing manufacturing cost. A two-layer PCB for an EMV chip card reader costs approximately USD 0.80 to manufacture; a four-layer PCB costs USD 2.50. With 100+ million POS terminals deployed globally, this difference represents hundreds of millions of dollars in hardware costs annually.

| Graph Concept | Syllabus | Industry | Application | Quantified Impact |
|---|---|---|---|---|
| Knowledge Graph | 7.1 Properties | OTT | Netflix recommendation engine | 80% of all viewing hours |
| Degree Centrality | 7.2 Degree | OTT | Trending content scoring | Real-time title ranking |
| Graph Connectivity | 7.3 Connectivity | OTT | Content coverage monitoring | Zero 'dead' titles |
| BFS Paths | 7.4 Path | Banking | Fraud chain tracing (k-hop) | 5-6x over rule systems (NVIDIA) |
| Eulerian Circuit | 7.7 Eulerian | Banking | Compliance audit scheduling | 100% coverage, 0% redundancy |
| Hamiltonian Path | 7.7 Hamiltonian | Banking | ATM cash logistics (SBI 68K) | Crores saved annually |
| Graph Colouring | 7.8 Colouring | FinTech/Banking | Compiler reg. allocation, scheduling | Visa: 65K txns/second |
| Four Colour Thm | 7.11 Four Colour | Banking | Geographic risk zone maps | 4 levels always sufficient |
| Decision Trees | 7.12 Rooted Trees | Banking | Credit scoring, loan approval | 95.79% ML accuracy |
| Binary Tree/B+ Tree | 7.13 Binary Tree | Banking | Database indexing | 3 lookups for 500M records |
| Merkle Tree | 7.13 Binary Tree | FinTech | Blockchain / CBDC | O(log N) tamper proof |
| Spanning Tree | 7.12 Trees | OTT | Netflix CDN topology | MST = optimal backbone |
| Sub-Graph | 7.5 Sub Graph | Banking | Fraud ring detection | GNN: 5-6x improvement |
| Isomorphism | 7.6 Isomorphism | FinTech | API schema matching | 70% integration effort saved |
| Planar Graph | 7.10 Planar | FinTech | PCB design (POS terminals) | 2-layer vs 4-layer cost saving |

# CHAPTER 8: DETAILED CASE STUDIES WITH DATA

## 8.1 Case Study: Netflix — End-to-End Recommendation Pipeline

Netflix's recommendation pipeline is one of the most sophisticated applied mathematics systems ever built. This case study traces a single user's evening session, showing exactly which discrete mathematical structure operates at each step.

| User Opens Netflix — Full Mathematical Journey |
|---|
| **STEP 1:** Set lookup: u ∈ Active_Subscribers? Check if user's subscription row exists (set membership O(1) hash) |
| ▼ |
| **STEP 2:** Function evaluation: f(u, region) → licensed_catalogue (function returns subset of titles for user's region) |
| ▼ |
| **STEP 3:** Graph traversal: BFS on knowledge graph from user's watched history nodes to candidate title nodes |
| ▼ |
| **STEP 4:** Ranking: Partial order on candidate titles by predicted engagement score (matrix factorisation output) |
| ▼ |
| **STEP 5:** Combinatorics: P(20,8) homepage layout — reinforcement learning selects best row ordering |
| ▼ |
| **STEP 6:** Truth table: SHOW(u,t) = active(u) ∧ licensed(t) ∧ ¬restricted(u,t) evaluated for each candidate |
| ▼ |
| **STEP 7:** Display: Top-ranked eligible titles shown; click events generate new graph edges in real time |

### 8.1.1 Netflix By The Numbers — Mathematical Scale

| Mathematical Operation | Daily Scale at Netflix | Discrete Math Tool | Latency Target |
|---|---|---|---|
| Set membership checks (auth) | 301.6M login events/day | Hash sets O(1) | <10ms |
| Function evaluations (eligibility) | 301.6M × 15K = 4.5T pairs | Boolean formula | <1ms (pre-indexed) |
| Graph traversal (KG queries) | Billions of edge traversals | BFS/DFS on knowledge graph | <50ms |
| Matrix factorisation (scoring) | Batch nightly + real-time | Linear algebra (ring ops) | <200ms |
| A/B test assignment | 1,000 experiments × 300M users | Combinatorial stratification | Real-time |

| Mathematical Operation | Daily Scale at Netflix | Discrete Math Tool | Latency Target |
|---|---|---|---|
| Thumbnail selection | 15K titles × 300M users | Multi-armed bandit | <100ms |
| Homepage construction | 300M unique pages/day | Permutation ranking | <200ms total |

**Netflix Research Papers on Discrete Math Applications**

• Steck et al. (2021) — 'Deep Learning for Recommender Systems: A Netflix Case Study' (AI Magazine)
Finding: Graph-based collaborative filtering outperforms matrix factorisation by 12-18% on cold-start users

• Gomez-Uribe & Hunt (2016) — 'The Netflix Recommender System' (ACM TOMS)
Finding: 75-80% of all streaming hours come from recommendations; saves $1B/year in churn reduction

• Aridor et al. (2025) — 'The Value of Personalized Recommendations: Evidence from Netflix' (arXiv:2511.07280)
Finding: Personalization increases consumer surplus by $2.9B/year; validates graph-based similarity

• Kamila, Prihandini & Agatha (2024) — 'Graph Theory Applications in Movie Recommendation Systems'
Finding: PageRank on user-content graph achieves precision@10 = 0.82, recall@10 = 0.74

## 8.2 Case Study: UPI — India's Mathematical Payment Miracle

India's Unified Payments Interface (UPI) is the world's most advanced real-time payment infrastructure. In 2025 it processed 228.3 billion transactions worth ₹299.7 lakh crore. December 2025 alone recorded 21.63 billion transactions — a world record for any real-time payment system. This achievement rests entirely on discrete mathematics.

| | | | |
|---|---|---|---|
| **228.3B** UPI Transactions in 2025 (NPCI) | **₹299.7Lcr** Total UPI Value 2025 | **21.63B** December 2025 record (single month) | **<1 sec** Average UPI settlement time |

### 8.2.1 UPI's Mathematical Architecture

| UPI Layer | Function | Discrete Math | Example |
|---|---|---|---|
| VPA Resolution | Phone → VPA address mapping | Bijective function f: ℕ → VPA | 9876543210 → user@oksbi |
| Bank Routing | Find shortest path to destination bank | Dijkstra on interbank graph (WOP) | NPCI routes via 3-hop path |
| Transaction ID | Unique ID for every payment | 128-bit UUID (Pigeonhole avoidance) | 550e8400-e29b-41d4-a716-... |

| UPI Layer | Function | Discrete Math | Example |
|---|---|---|---|
| Digital Signature | Authenticate each payment | ECC group operation (Abelian group) | ECC-256 sign on TEE chip |
| Fraud Scoring | Flag suspicious transactions | Boolean rule engine + ML trees | Rule: amt>50K∧new_device→flag |
| Netting (end-of-day) | Settle bank-to-bank obligations | GCD/Euclidean (reduce transactions) | ₹300M - ₹200M net = ₹100M |
| Audit Trail | Record all transactions immutably | Merkle tree (Binary tree) | Hash chain of daily batches |

## 8.3 Case Study: Fraud Detection — Graph Neural Networks at Scale

Financial fraud costs the global economy over $485 billion annually. Traditional rule-based systems catch known fraud patterns but miss novel ones. Graph Neural Networks (GNNs), built on graph theory (Unit 5), are transforming this landscape.

### 8.3.1 How GNNs Detect Fraud — Step by Step

**GNN Fraud Detection Pipeline (JPMorgan, HSBC, NVIDIA Research 2024)**

INPUT: Transaction graph G = (Accounts, Transactions, Features)
→ Nodes: Every bank account (millions of vertices)
→ Edges: Every transaction (billions of edges, directed, weighted by amount)
→ Node features: Account age, transaction velocity, device count, geographic spread
→ Edge features: Amount, timestamp, merchant category, location

STEP 1 — Message Passing (Graph Convolution):
Each node aggregates features from its k-hop neighbourhood
$h_v^{(k)} = \sigma(W^{(k)} \cdot AGG(\{h_u^{(k-1)} : u \in N(v)\}))$
This is a ring operation on the feature space (matrix multiplication)

STEP 2 — Community Detection (Sub-Graph Analysis):
Louvain algorithm identifies dense sub-graphs (fraud rings)
Modularity Q > 0.4 signals a suspicious community

STEP 3 — Node Classification:
Each account node receives a fraud probability score [0,1]
Threshold at 0.85: accounts above this are flagged for review

RESULT: 5-6x improvement over rule-based systems (NVIDIA, 2024)

| Model Type | Accuracy | Precision | Recall | F1 Score | Source |
|---|---|---|---|---|---|
| Random Forest (baseline) | 95.79% | 94.2% | 93.8% | 94.0% | Cogent B&M 2025 |
| Gradient Boosting (XGBoost) | 94.3% | 93.1% | 92.7% | 92.9% | JRFM MDPI 2025 |

| Model Type | Accuracy | Precision | Recall | F1 Score | Source |
|---|---|---|---|---|---|
| GNN (GraphSAGE) | 97.2% | 96.8% | 95.9% | 96.4% | NVIDIA Research 2024 |
| GNN + Autoencoder hybrid | 98.1% | 97.4% | 96.8% | 97.1% | IEEE Access 2025 |
| DyHDGE (temporal graph) | 98.6% | 97.9% | 97.4% | 97.6% | SAGE Journals 2025 |
| Traditional rule engine | ~80% | ~75% | ~70% | ~72% | Industry average |

## 8.4 Case Study: Blockchain and Cryptographic Structures in FinTech

Blockchain technology, valued at USD 9.95 billion in the FinTech sector (2024), is entirely constructed from discrete mathematical structures: Merkle trees (binary trees), cryptographic hash functions (modular arithmetic), digital signatures (group theory), and consensus algorithms (graph theory).

### 8.4.1 Merkle Tree Structure in a Bitcoin Block

**Anatomy of a Bitcoin Block — Discrete Math at Every Layer**

BLOCK HEADER:
- Previous block hash: SHA-256(prev_block) — modular arithmetic on 256-bit integers
- Merkle Root: Root of binary tree of all transaction hashes
- Nonce: Number found by mining (existential quantifier: ∃ nonce: hash < target)

MERKLE TREE (Syllabus: Binary Tree — 7.13):
Level 0 (Leaves): Hash of each transaction: H(Tx1), H(Tx2), ..., H(TxN)
Level 1: H(H(Tx1)+H(Tx2)), H(H(Tx3)+H(Tx4)), ...
Level k (Root): Single 256-bit root hash
Height: $\log_2(N)$ — for 2048 transactions: height = 11 levels
Proof size: O(log N) hashes — efficient membership verification

DIGITAL SIGNATURE (Syllabus: Groups — 4.4):
Private key: random integer d ∈ [1, n-1] (n = group order)
Public key: Q = d·G (scalar multiplication on elliptic curve group)
Sign: (r,s) = ECDSA(d, transaction_hash)
Verify: Check if (r,s) valid under Q — group membership test

CONSENSUS (Syllabus: Graph Connectivity — 7.3):
Longest chain rule: miners maintain the connected chain with most computational work
51% attack requires controlling majority of graph's computational nodes

# CHAPTER 9: VISUAL SUMMARY — CONCEPT-INDUSTRY MAPPING

## 9.1 Complete Syllabus Coverage Matrix

The following matrix maps every syllabus topic to its industrial application, providing a complete reference for the reader:

| Syllabus Topic | OTT Platform Application | Banking Application | FinTech Application |
|---|---|---|---|
| Operations and Laws of Sets | Content library set ops for licensing | Account set operations for product offers | Wallet balance set reconciliation |
| Cartesian Products | User × Content scoring pairs | Account × Product feature matrix | Customer × Transaction pairs |
| Binary Relation | Watch history relations | Authorisation relation (who can pay whom) | KYC document-customer relation |
| Partial Ordering Relations | Content ranking by engagement | Product tier hierarchy | Risk score ordering |
| Equivalence Relation | Content genre clustering | Customer segment partitioning | Transaction category grouping |
| Image of a Set | 'Users who also watched' image | AML: image of suspicious accounts | Fund tracing: f(flagged_accounts) |
| Bijective Functions | User ID ↔ Profile mapping | Account ↔ Card number (1-to-1) | Tokenization: Card → Token |
| Inverse Functions | Reverse recommendation lookup | Decryption: ciphertext → plaintext | Token → Card (settlement) |
| Composite Functions | Multi-step recommendation pipeline | Multi-layer auth: device→session→OTP | Multi-hop payment routing |
| Size of a Set / Cardinality | Catalogue size: 15,000+ titles | Number of accounts, products | Transaction volume counting |
| Countable / Uncountable Sets | Finite catalogue, infinite user prefs | Discrete transactions, continuous risk | Finite token space analysis |
| Cantor's Diagonal Argument | Infinite personalisation space | Proves limits of rule sets for fraud | Stream processing impossibility |
| Power Set Theorem | Feature power set: 2^n user profiles | Product combination offers | Risk factor combinations |
| Schroeder–Bernstein Theorem | Platform catalogue merge (Disney+JioCinema) | Bank merger customer DB merge | FinTech acquisition data merge |
| Basic Counting Technique | A/B test variant counting | Audit coverage counting | Fee schedule combinations |
| Inclusion-Exclusion | Unique viewer counting across genres | Unique customer counting across products | Transaction deduplication |

| Syllabus Topic | OTT Platform Application | Banking Application | FinTech Application |
|---|---|---|---|
| Pigeonhole Principle | Collision-free experiment assignment | UUID/transaction ID uniqueness | Nonce space in proof-of-work |
| Permutation and Combination | Homepage row ordering | PIN/OTP combination security | Trade order sequence optimisation |
| Mathematical Induction | Gradient descent convergence proof | EMI formula correctness | Smart contract loop invariant |
| Well Ordering Principle | Optimal homepage layout exists | Optimal fraud route detection | Optimal payment routing (Dijkstra) |
| Recursive Definition | Recommendation tree recursion | Compound interest recursion | Merkle tree recursive construction |
| Division Algorithm | Timestamp modular partition | Modular arithmetic in hashing | Remainder in currency conversion |
| Prime Numbers | — | RSA prime key generation | ECC prime field parameters |
| GCD / Euclidean Algorithm | — | Modular inverse in RSA | Bilateral netting optimisation |
| Fundamental Theorem of Arithmetic | — | RSA uniqueness of factorisation | Blockchain hash uniqueness |
| Basic Connectives / Truth Tables | Content eligibility 8-row truth table | Fraud condition truth table | Payment validity truth table |
| Logical Equivalence (De Morgan) | Query optimisation (¬B∧¬R ≡ ¬(B∨R)) | Rule engine simplification | API validation simplification |
| Logical Implication | Engagement prediction: history→preference | Geolocation anomaly detection | Transaction threshold implications |
| Rules of Inference | Recommendation chain (MP, HS) | Fraud inference (MP, MT, HS) | Compliance rule chaining |
| Quantifiers (∀, ∃) | ∀user: show top-10 per category | ∀txn: amount > threshold → flag | ∃merchant: new ∧ amount > 5×avg |
| Forward Proof | Algorithm correctness | RSA decryption correctness | Hash collision resistance |
| Proof by Contradiction | No perfect compressor exists | Impossibility of zero-cost security | Impossibility of instant+secure+free |
| Proof by Contrapositive | Non-recommended = non-relevant | Non-2FA = non-sensitive endpoint | Non-compliant = non-approved |
| Necessity and Sufficiency | Engagement necessary for retention | CIBIL ≥ 750 necessary for loan | KYC necessary for account opening |
| Algebraic Structures (1 op) | Log pipeline (semigroup: concat) | Transaction chain (monoid) | Payment message (free monoid) |
| Semi-groups / Monoids | Log aggregation pipelines | Batch transaction processing | ISO 20022 message composition |
| Groups | Symmetry in content layout | RSA multiplicative group mod n | ECC abelian group, crypto keys |

| Syllabus Topic | OTT Platform Application | Banking Application | FinTech Application |
|---|---|---|---|
| Congruence Relations | Content hash partitioning | Modular account segmentation | Consistent hashing in distributed DBs |
| Cyclic Groups / Free Groups | — | RSA cyclic group structure | Schnorr signature group |
| Permutation Groups | UI layout permutations | API token rotation (OAuth) | Key rotation schemes |
| Rings | Neural net weight matrices | Exact integer bank arithmetic | Polynomial rings in lattice crypto |
| Integral Domain / Fields | Feature scaling (real field) | No rounding errors: $\mathbb{Z}$ domain | Currency arithmetic in $\mathbb{Z}$ |
| Boolean Algebra | Content filtering rules | Fraud detection rule engine | ASIC circuit optimisation |
| DNF/CNF and K-maps | UI decision logic SOP | Fraud circuit chip design | POS terminal ASIC K-map |
| Graph Properties | Knowledge graph structure | Transaction graph properties | Blockchain graph properties |
| Degree Centrality | Title popularity scoring | Account centrality in fraud ring | Address activity in blockchain |
| Graph Connectivity | Catalogue coverage monitoring | AML chain coverage | Network partition tolerance |
| Paths and Cycles | 'Because you watched' path | Fraud chain: source→mule→exit | Payment routing paths |
| Sub-Graphs | Genre sub-graphs | Fraud ring dense sub-graphs | Exchange cluster sub-graphs |
| Graph Isomorphism | Schema matching between catalogues | API schema compatibility check | Cross-chain protocol matching |
| Eulerian Walks | Content tour (all genres once) | Audit sequence design | Transaction sweep coverage |
| Hamiltonian Walks | — | ATM cash logistics (SBI 68K) | Delivery route optimisation |
| Graph Colouring | Schedule simultaneous A/B tests | RBI report batch scheduling | Compiler register allocation |
| Colouring Maps | Regional content licensing zones | Geographic fraud risk zones | Regulatory jurisdiction maps |
| Planar Graphs | — | Circuit board design (ATM cards) | POS terminal PCB layout |
| Four Colour Theorem | — | Risk zone map (4 colours sufficient) | Geofenced service zone design |
| Trees and Rooted Trees | Decision tree recommendations | Approval hierarchy trees | Asset hierarchy in portfolios |
| Binary Tree | Binary search in content metadata | B+ tree database indexing | Merkle trees in blockchain |

## 9.2 Research Papers Summary Table

The following table summarises all key research papers cited in this assignment, along with their key quantitative findings:

| Paper | Year | Journal | Key Quantitative Finding | Industry |
|---|---|---|---|---|
| Comparative ML for Banking Fraud (565K txns) | 2025 | Cogent Business & Management | Random Forest: 95.79% accuracy; best among 6 models | Banking |
| GNNs for Financial Fraud Detection: Review | 2024 | arXiv (2411.05815) | GNNs outperform traditional methods by 5-6x in fraud ring detection | Banking/FinTech |
| GNNs and Autoencoders for Fraud | 2025 | IEEE Access Vol.13 | Precision-recall balance improved; real-time credit card fraud prevention | Banking |
| Graph Learning for Financial Fraud | 2025 | Intelligent Computing (AAAS) | Semi-GNN with unlabeled data: best F1 for transaction graph fraud | Banking |
| DyHDGE Temporal Transaction Graphs | 2025 | SAGE Journals | Dynamic graph embedding: state-of-the-art temporal fraud detection | Banking |
| Graph Theory in Movie Recommendations | 2024 | ResearchGate (Conference) | PageRank on user-content graph: Precision@10=0.82, Recall@10=0.74 | OTT |
| Netflix Recommender System | 2016 | ACM TOMS (Vol.6 No.4) | 75-80% views from recommendations; $1B/year savings from churn reduction | OTT |
| Value of Personalised Recommendations | 2025 | arXiv:2511.07280 | Personalization increases consumer surplus by $2.9B/year | OTT |
| Deep Learning for Recommender Systems | 2021 | AI Magazine (Vol.42 No.3) | Graph-based CF outperforms MF by 12-18% on cold-start users | OTT |
| Hybrid DBN + Quantum for Fraud | 2025 | Discover Applied Sciences | Quantum-enhanced GNN improves training speed + accuracy for fraud | FinTech |
| ML as Tool for Fraud Risk Management | 2025 | JRFM MDPI (Vol.18 No.3) | XGBoost on 1.5M txns: 94.3% accuracy; reviews ML evolution in banking | Banking |

## 9.3 Market Data Reference Table

| Metric | Value | Year | Source |
|---|---|---|---|
| Global OTT Revenue | USD 316 billion | 2024 | Statista Global Outlook |
| OTT Revenue Projection | USD 595 billion | 2030 | MarkNtel Advisors (CAGR 16.7%) |
| OTT Revenue Projection | USD 1,425.63 billion | 2033 | Grand View Research (CAGR 18.6%) |
| OTT Revenue Projection | USD 2,816.85 billion | 2034 | Precedence Research (CAGR 24.42%) |
| Global OTT Users | 3.71 billion | 2024 | VPlayed / Industry Reports |
| Netflix Subscribers | 301.6 million | Q3 2025 | Netflix Investor Relations |
| Netflix Q1 Revenue | USD 10.54 billion | 2025 | Netflix Investor Relations |
| Netflix Views from Recommendations | 75–80% | 2024 | Gomez-Uribe & Hunt (Netflix) |
| Netflix Churn Savings from AI | USD 1 billion+/year | 2024 | Netflix Technology Blog |
| Global FinTech Market | USD 228.02 billion | 2024 | Zion Market Research |
| Global FinTech Market | USD 394.88 billion | 2025 | Fortune Business Insights |
| FinTech Projection 2034 | USD 1,760.18 billion | 2034 | Fortune Business Insights (CAGR 18.2%) |
| FinTech Investment | USD 43.5 billion | 2024 | Fortunly Report 2025 |
| FinTech Unicorns | 414 companies | Jan 2025 | Fortunly Report 2025 |
| UPI Transactions (Annual) | 228.3 billion | 2025 | NPCI Annual Data |
| UPI Transaction Value | ₹299.7 lakh crore (~$3.6T) | 2025 | NPCI Annual Data |
| UPI Record Month | 21.63 billion transactions | December 2025 | NPCI Monthly Report |
| UPI Transaction Volume (Oct 2024) | 16.58 billion | Oct 2024 | NPCI Monthly Report |
| Global Digital Wallet Txn Value | USD 20.37 trillion | 2025 | Industry Reports |
| Blockchain in FinTech Market | USD 9.95 billion | 2024 | Mordor Intelligence |
| Blockchain FinTech Projection | USD 46.87 billion | 2030 | Mordor Intelligence (CAGR 46.92%) |
| Banking NPAs (India) | ₹2.09 lakh crore | FY2024 | RBI Annual Report |
| Banking Fraud Losses (India) | ₹13,930 crore | FY2023-24 | RBI Annual Report |
| Global Fraud Losses (Projected) | USD 403 billion | 10-year horizon | Nilson Report 2024 |
| SBI ATM Network | 68,000+ ATMs | 2024 | SBI Annual Report |
| SBI Account Holders | 500 million+ | 2024 | SBI Annual Report |
| Visa Peak Transactions | 65,000 per second | 2024 | Visa Inc. Fact Sheet |
| ML Fraud Detection Accuracy | 95.79% (Random Forest) | 2025 | Cogent Business & Management |
| GNN vs Rules Improvement | 5-6x better detection | 2024 | NVIDIA Research |

# CHAPTER 10: STEP-BY-STEP WORKED EXAMPLES

This chapter provides fully worked numerical examples for the most important discrete mathematics concepts applied in industry. Each example mirrors what happens inside real production systems at Netflix, UPI, or Indian banks.

## 10.1 Worked Example: Set Operations for Netflix Recommendation

### Problem Setup

Netflix has the following user watching data for a single day. Compute the correct recommendation set for each scenario.

| User Set | Genre Preferences | Description |
|---|---|---|
| A = {u1, u2, u3, u4, u5} | Action, Thriller | Users who watched action content |
| B = {u3, u4, u6, u7} | Drama, Romance | Users who watched drama content |
| C = {u1, u5, u8} | Documentary | Users who watched documentary content |
| U = {u1...u10} | All genres | Universal set: all 10 users this session |

**Solution — Set Operations Step by Step**

1) A ∩ B = {u3, u4} → These users like both action AND drama → Show action-drama hybrid titles
2) A ∪ B = {u1,u2,u3,u4,u5,u6,u7} → 7 users engaged with action OR drama → Broad action/drama promos
3) A \ B = {u1, u2, u5} → Action-only watchers (not drama) → Recommend pure action sequels
4) A' (complement) = {u6,u7,u8,u9,u10} → Non-action watchers → Definitely NOT show action in top row
5) |P(A)| = 2^|A| = 2^5 = 32 → 32 possible sub-segments within action fans for micro-targeting
6) A × B = 5×4 = 20 ordered pairs → 20 (user,genre) combinations for cross-recommendation
7) Verification: |A ∪ B| = |A| + |B| - |A ∩ B| = 5 + 4 - 2 = 7 ✓ (Inclusion-Exclusion check)

## 10.2 Worked Example: RSA Encryption for Banking (Simplified)

This example uses small primes to illustrate RSA — in production, 2048-bit primes are used, but the mathematics is identical.

**RSA Worked Example — Step by Step**

STEP 1: Choose primes p = 61, q = 53
STEP 2: Compute n = p × q = 61 × 53 = 3,233 (this is the public modulus)
STEP 3: Compute φ(n) = (p-1)(q-1) = 60 × 52 = 3,120 (Euler's totient)
STEP 4: Choose e = 17 (must satisfy: gcd(17, 3120) = 1 — use Euclidean Algorithm)
Euclidean verification: gcd(17, 3120):
3120 = 183×17 + 9 → 17 = 1×9 + 8 → 9 = 1×8 + 1 → 8 = 8×1 + 0
GCD = 1 ✓ (e=17 is valid)
STEP 5: Find d such that 17d ≡ 1 (mod 3120) → d = 2753

Verify: 17 × 2753 = 46,801 = 15 × 3120 + 1 ✓
STEP 6: Public Key = (n=3233, e=17) | Private Key = (n=3233, d=2753)
STEP 7: Encrypt message M = 65 (e.g., ASCII 'A' representing transaction code):
C = M^e mod n = 65^17 mod 3233 = 2790
STEP 8: Decrypt ciphertext C = 2790:
M = C^d mod n = 2790^2753 mod 3233 = 65 ✓ (original message recovered)
RESULT: A bank sends encrypted transaction data '2790' over the wire; only recipient with d=2753 can read it

## 10.3 Worked Example: Dijkstra's Algorithm for UPI Payment Routing

NPCI routes UPI payments through a network of banks. Find the shortest (lowest cost) path from user's bank SBI to destination bank HDFC.

**Network Setup**

Banks (Vertices): SBI(S), PNB(P), Axis(A), ICICI(I), HDFC(H), Kotak(K)
Interbank links (Edges) with routing costs (ms latency):
S–P: 5ms | S–A: 10ms | P–I: 3ms | A–I: 7ms | I–H: 4ms | A–H: 8ms | P–K: 6ms | K–H: 9ms

**Dijkstra's Algorithm — Iteration Table**

Initial: dist={S:0, P:∞, A:∞, I:∞, H:∞, K:∞} | Visited={}

Iteration 1: Visit S (min dist=0)
Update neighbours: P=min(∞,0+5)=5, A=min(∞,0+10)=10
dist={S:0, P:5, A:10, I:∞, H:∞, K:∞}

Iteration 2: Visit P (min unvisited dist=5)
Update: I=min(∞,5+3)=8, K=min(∞,5+6)=11
dist={S:0, P:5, A:10, I:8, H:∞, K:11}

Iteration 3: Visit I (min unvisited dist=8)
Update: H=min(∞,8+4)=12
dist={S:0, P:5, A:10, I:8, H:12, K:11}

Iteration 4: Visit A (min unvisited dist=10)
Update: H=min(12,10+8)=12 (no improvement)

Iteration 5: Visit K (dist=11): H=min(12,11+9)=12 (no improvement)

RESULT: Shortest path S→P→I→H = 5+3+4 = 12ms
NPCI routes this UPI payment: SBI → PNB → ICICI → HDFC in 12ms total
Well Ordering Principle guarantees this minimum exists (dist set is non-empty positive integers)

## 10.4 Worked Example: Boolean Algebra Fraud Rule Simplification

A bank's fraud team writes the following raw fraud detection rule. Simplify it using Boolean algebra laws to reduce computation time.

**Original Fraud Rule (Unsimplified)**

**F = (A∧B∧C) ∨ (A∧B∧¬C) ∨ (A∧¬B∧D)**

Where: A=amount>50K, B=night_time, C=known_device, D=international_IP

**Boolean Algebra Simplification — Step by Step**

F = (A∧B∧C) ∨ (A∧B∧¬C) ∨ (A∧¬B∧D)

Step 1: Factor out (A∧B) from first two terms:
F = (A∧B)∧(C ∨ ¬C) ∨ (A∧¬B∧D) [Distributive Law]

Step 2: Apply Complement Law: C ∨ ¬C = TRUE (Tautology)
F = (A∧B)∧TRUE ∨ (A∧¬B∧D) [Complement Law]

Step 3: Apply Identity Law: X∧TRUE = X
F = (A∧B) ∨ (A∧¬B∧D) [Identity Law]

Step 4: Factor out A:
F = A ∧ (B ∨ (¬B∧D)) [Distributive Law]

Step 5: Apply Absorption variant: B ∨ (¬B∧D) = B ∨ D
F = A ∧ (B ∨ D) [Simplified!]

RESULT: F = A ∧ (B ∨ D) = 'amount>50K AND (night_time OR international_IP)'
Original: 3 AND-gates, 2 OR-gates, 1 NOT-gate = 6 operations per transaction
Simplified: 1 AND-gate, 1 OR-gate = 2 operations per transaction
Speedup: 3× faster — critical at Visa's 65,000 transactions/second

## 10.5 Worked Example: Mathematical Induction for EMI Formula

Prove by Mathematical Induction that the outstanding loan balance after n EMI payments is: B(n) = B(0)·(1+r)^n - P·[(1+r)^n - 1]/r

**Proof by Mathematical Induction**

GIVEN: B(n) = B(n-1)·(1+r) - P (recurrence relation for monthly balance)
CLAIM: B(n) = B(0)·(1+r)^n - P·[(1+r)^n - 1]/r for all n ≥ 1

BASE CASE (n=1):
B(1) = B(0)·(1+r) - P [from recurrence]
RHS: B(0)·(1+r)^1 - P·[(1+r)^1 - 1]/r
= B(0)·(1+r) - P·[r/r]

= B(0)·(1+r) - P ✓ Base case holds.

INDUCTIVE STEP: Assume B(k) = B(0)·(1+r)^k - P·[(1+r)^k - 1]/r (Inductive Hypothesis)
Show: B(k+1) = B(0)·(1+r)^(k+1) - P·[(1+r)^(k+1) - 1]/r

B(k+1) = B(k)·(1+r) - P [recurrence]
= [B(0)·(1+r)^k - P·((1+r)^k-1)/r]·(1+r) - P [IH]
= B(0)·(1+r)^(k+1) - P·(1+r)·((1+r)^k-1)/r - P
= B(0)·(1+r)^(k+1) - P·[(1+r)^(k+1)-(1+r)]/r - P
= B(0)·(1+r)^(k+1) - P·[(1+r)^(k+1)-(1+r)+r]/r
= B(0)·(1+r)^(k+1) - P·[(1+r)^(k+1) - 1]/r ✓

CONCLUSION: By Mathematical Induction, B(n) = B(0)·(1+r)^n - P·[(1+r)^n-1]/r for all n≥1.
This formula is used by every bank's loan management system in India and globally.

## 10.6 Worked Example: Graph Colouring for Report Scheduling

A bank has 6 regulatory reports. Some pairs conflict (share database resources). Find minimum time slots using graph colouring.

**Graph Colouring — Report Scheduling**

Reports: R1(Daily Balance), R2(Fraud Summary), R3(AML Report), R4(CRAR), R5(Liquidity), R6(Credit Risk)
Conflicts (edges — shared database tables):
R1-R2 (both read transactions table)
R1-R3 (both read accounts table)
R2-R4 (both read risk scores table)
R3-R5 (both read liquidity data)
R4-R6 (both read credit portfolio)
R5-R6 (both read asset tables)

Greedy Graph Colouring (Degree-ordered):
Degrees: R1=2, R2=2, R3=2, R4=2, R5=2, R6=2 (all equal, proceed in order)
R1 → Colour 1 (Time Slot 1)
R2 → R2 conflicts with R1 → Colour 2 (Time Slot 2)
R3 → R3 conflicts with R1 → Colour 2 (Time Slot 2) [no conflict with R2]
R4 → R4 conflicts with R2 → Colour 1 or 3 → Colour 1 (Time Slot 1) [no conflict with R1]
Wait: R4 conflicts with R2(Slot 2), so try Slot 1: R4 and R1 don't conflict → Slot 1 ✓
R5 → conflicts R3(Slot 2): try Slot 1: conflicts R1? No → Slot 1
R5 conflicts R3(Slot 2) and R6 not assigned yet → Slot 1 ✓
R6 → conflicts R4(Slot 1), R5(Slot 1) → must use Slot 2: conflicts R2? No, R3? No → Slot 2 ✓... wait conflicts R5 and R4
R6 conflicts R4(Slot1), R5(Slot1) → try Slot 3 → no conflicts → Slot 3

FINAL SCHEDULE: Slot 1: {R1, R4, R5} | Slot 2: {R2, R3} | Slot 3: {R6}
χ(G) = 3 → Only 3 time slots needed instead of 6 sequential! 50% time saving.

# CHAPTER 11: CONCLUSION, FUTURE SCOPE & SUMMARY

## 11.1 Summary of All Unit-Industry Connections

This assignment has demonstrated, with research citations, market data, and worked examples, that every unit of the Discrete Mathematics syllabus has deep, active, and economically transformative applications across OTT platforms, banking systems, and FinTech. The table below provides a consolidated master summary:

| Unit | Core Concept | OTT Application | Banking Application | FinTech Application |
|---|---|---|---|---|
| Unit 1 | Sets & Functions | User segmentation, content eligibility | Account hierarchies, AML tracing | Tokenization bijections, DB mergers |
| Unit 1 | Relations & Orders | Equivalence-based content clustering | Customer risk partition, auth graphs | VPA function, schema mappings |
| Unit 2 | Counting & PnC | A/B testing, homepage permutations | UUID collision avoidance, audit | Routing path enumeration, prime security |
| Unit 2 | Induction & WOP | Algorithm correctness proofs | EMI formula, convergence proofs | Payment routing optimality guarantee |
| Unit 3 | Logic & Inference | Eligibility truth tables, De Morgan | Modus Ponens fraud rules, MT login | LLM guardrails, loan N&S conditions |
| Unit 3 | Proof Techniques | Impossibility of perfect compression | RSA correctness, contradictions | RSA security, NP-completeness |
| Unit 4 | Groups | — | RSA encryption, ECC signatures | Blockchain group law, token rotation |
| Unit 4 | Rings & Fields | Neural network computation | Exact bank arithmetic | Currency in $\mathbb{Z}$, homomorphic encryption |
| Unit 4 | Boolean Algebra | Content filtering | Fraud rule engine | ASIC K-map optimisation |
| Unit 5 | Graphs | Knowledge graph, CDN MST | Transaction fraud detection | Merkle trees, blockchain ledgers |
| Unit 5 | Trees | Decision tree recommendations | B+ tree DB index, org hierarchy | CBDC Merkle, routing trees |
| Unit 5 | Graph Colouring | — | Audit slot scheduling | Compiler register allocation |

## 11.2 Key Statistics Summary

| | | | |
|---|---|---|---|
| **$316B** OTT Market 2024 (Statista) | **$228B** FinTech Market 2024 | **228B** UPI Transactions 2025 (NPCI) | **95.79%** ML Fraud Detection Accuracy |

| 301.6M | 3.71B | $43.5B | ₹2.09Lcr |
|---|---|---|---|
| Netflix Subscribers Q3 2025 | Global OTT Users 2024 | FinTech Investment 2024 | Banking NPAs India FY2024 |

## 11.3 Future Scope

**Emerging Applications of Discrete Math (2025–2035)**

POST-QUANTUM CRYPTOGRAPHY: Shor's algorithm (group theory) will break RSA/ECC on quantum computers.
→ Banks migrating to NIST-standardised lattice-based cryptography (polynomial rings over finite fields)

HOMOMORPHIC ENCRYPTION: Ring homomorphisms enabling computation on encrypted financial data.
→ Privacy-preserving credit scoring; no raw data exposure; deployed by Microsoft Azure, Google Cloud

GRAPH NEURAL NETWORKS (GNNs): Replace rule-based fraud systems by 2027.
→ JPMorgan, HSBC, SBI deploying GNNs on transaction graphs; 5-6x improvement over rules

FORMAL VERIFICATION: Prove smart contracts correct using propositional/predicate logic.
→ Essential for DeFi and RBI CBDC systems; prevents billion-dollar smart contract bugs

QUANTUM MACHINE LEARNING: Quantum graph algorithms for portfolio optimisation.
→ Quantum walks on financial graphs; polynomial speedup over classical algorithms

HYPERGRAPH MODELS: Extend graph theory to model group transactions.
→ Split bills, consortium loans, family accounts modelled as hyperedges

## 11.4 Final Reflection

A student who masters Sets and Functions will design better databases. One who masters Counting will understand algorithm complexity. One who masters Logic will build correct and secure software. One who masters Algebraic Structures will understand the mathematics of privacy and security. One who masters Graph Theory will understand the internet, social networks, payment systems, and recommendation engines.

The industries explored in this assignment — OTT, Banking, FinTech — collectively represent over USD 1 trillion in annual economic value, serve more than 4 billion users, and are growing at double-digit rates. Their engineers are, whether they know it or not, applied discrete mathematicians. The tools they build today will reshape the global economy over the next decade.

# REFERENCES

## Published Research Papers


**Comparative analysis of machine learning models for detection of fraudulent banking transactions**

Authors: Multiple Authors

Journal: Cogent Business & Management (2025 (Mar 8))

Key Finding: Using 565,000 real banking transactions, Random Forest achieved 95.79% accuracy — highest among tested models. DOI: 10.1080/23311975.2025.2474209

**Graph Neural Networks for Financial Fraud Detection: A Review**

Authors: Dawei Cheng et al.

Journal: arXiv:2411.05815 (November 2024)

Key Finding: Proposes unified GNN framework for financial fraud detection. GNNs significantly outperform traditional methods in capturing relational patterns in financial networks.

**Enhancing Fraud Detection in Banking with Deep Learning: GNNs and Autoencoders**

Authors: Alarfaj FK, Shahzadi S.

Journal: IEEE Access (2025 (Vol. 13))

Key Finding: GNN and autoencoder hybrid achieves balanced precision-recall for real-time credit card fraud prevention. DOI: 10.1109/ACCESS.2024.3466288

**Graph Learning-Empowered Financial Fraud Detection: Progress and Future Directions**

Authors: Li Y et al.

Journal: Intelligent Computing (Science/AAAS) (2025)

Key Finding: Graph learning with relational structure enhances fraud detection. Covers GNN methods for AML, credit risk, and transaction fraud.

**Real-time Transaction Flow Analysis with Graph Neural Networks**

Authors: Xueran Wang et al.

Journal: SAGE Journals / SAGE Publications (2025)

Key Finding: DyHDGE dynamic heterogeneous transaction graph embedding for safety-centric fraud detection. Extends to temporal graph dynamics.

**Machine Learning as a Tool for Assessment and Management of Fraud Risk**

Authors: Multiple Authors

Journal: JRFM — MDPI (2025 (Vol. 18, No. 3))

Key Finding: Tested CART, Gradient Boosting, XGBoost on 1.5M transactions. Reviews evolution of ML for banking fraud risk management.

**Graph Theory: Applications in Movie Recommendation Systems (Netflix)**

Authors: Kamila IC, Prihandini RM, Agatha AB

Journal: ResearchGate / Published Conference (June 2024)

Key Finding: Films and user preferences as nodes in a graph; edges connecting users to watched/rated films. PageRank applied for recommendation scoring.

**The Netflix Recommender System: Algorithms, Business Value, and Innovation**

Authors: Gomez-Uribe CA, Hunt N

Journal: ACM Trans. Mgmt. Inf. Syst. (2016 (Vol. 6, No. 4))

Key Finding: Netflix's recommendation system saves $1B/year; 75–80% of views from algorithmic recommendations. Foundational industry paper.

## Market Research & Industry Reports


1. Statista (2024). OTT Video — Worldwide Market Forecast. Revenue in OTT Video market: US$316B (2024) → US$351B (2025). statista.com/outlook/amo/media/tv-video/ott-video/worldwide
2. Grand View Research (2024). Over the Top Devices and Services Market Report 2033. USD 298.84B (2024) → USD 1,425.63B (2033). grandviewresearch.com
3. Precedence Research (2025). Over the Top (OTT) Market Size to Surpass USD 2816.85 Bn by 2034. CAGR 24.42%. precedenceresearch.com
4. Fortune Business Insights (2025). FinTech Market Overview. USD 394.88B (2025) → USD 1,760.18B (2034). CAGR 18.20%.
5. Zion Market Research (2024). Fintech Market Size, Trend, Growth. USD 228.02B (2024) → USD 727.17B (2034). CAGR 15.60%.
6. Mordor Intelligence (2024). Fintech Blockchain Market. USD 6.85B (2025) → USD 46.87B (2030). CAGR 46.92%.
7. Fortunly (2025). 20+ Essential Fintech Statistics for 2025. USD 43.5B investment 2024; 414 unicorns Jan 2025.
8. CoinLaw.io (2026). UPI Statistics 2026: Insights and Trends. Annual UPI: 228.3B transactions (2025). ₹299.7 lakh crore value.
9. Uscreen (2025). Top OTT Statistics and Trends. 99% US households, avg 2.9 subscriptions, ARPU $83.28 (2025).
10. VPlayed (2025). OTT Statistics 2025. 3.71B global OTT users 2024; India cricket 59M peak concurrent viewers.
11. MarkNtel Advisors (2024). Global OTT Platform Market. USD 235B (2024) → USD 595B (2030). CAGR 16.7%.


## Textbooks and Standards


1. Rosen, K.H. (2019). Discrete Mathematics and Its Applications (8th ed.). McGraw-Hill Education.
2. Cormen, T.H. et al. (2022). Introduction to Algorithms (4th ed.). MIT Press.
3. Stinson, D.R. (2018). Cryptography: Theory and Practice (4th ed.). CRC Press.
4. Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press. deeplearningbook.org
5. NIST (2024). Post-Quantum Cryptography Standardisation. FIPS 203, 204, 205. csrc.nist.gov
6. Reserve Bank of India (2024). Annual Report on Banking Fraud Statistics FY 2023-24. rbi.org.in
7. NPCI (2025). UPI Annual Report and Transaction Data 2024-25. npci.org.in
8. ISO 20022 (2023). Universal Financial Industry Message Scheme. iso20022.or


# Thank You!